4-30-2008

Application of Term-Rewriting Grammar in Chemical Reaction Prediction

Patra Volarath

APPLICATION OF TERM-REWRITING GRAMMAR IN CHEMICAL REACTION PREDICTION

by

PATRA VOLARATH

Under the Direction of Dr. Robert Harrison

ABSTRACT

Synthesis planning is a critical process in chemical design. A number of computer programs have been developed to assist the chemists with this procedure. Most of the programs utilize combinations of computational approaches. These have been successfully applied to a number of the synthesis predictions. However, they require numerous rules to screen for potential targets, as well as to keep the system from reaching the combinatorial explosion. This results in the advanced algorithms becoming more complex and parameter-sensitive. This can be problematic, particularly in the cases in which a large number of the compounds are to be handled, because it can not only result in a lengthy computational time, but also cause some of the highly potential targets to be missed.

We developed a simpler approach for the reaction prediction using a term-rewriting grammar. The term-rewriting strategy is used to directly assign reactions to the compounds. This greatly reduces the number of rules that are usually required for these steps, and, hence,

facilitates the prediction performance, while maintaining the prediction accuracy. In this dissertation, the designs of the developed algorithms and their results are first being presented, followed by a discussion of the approach's application in the chemical design in the final chapter.

INDEX WORDS: Term-rewriting programming, Maude, Chemical Design, Reaction Prediction

DESIGNING OF A GRAMMATICAL APPROACH FOR CHEMICAL REACTION PREDICTION

by

PATRA VOLARATH

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2008

DESIGNING OF A GRAMMATICAL APPROACH FOR CHEMICAL REACTION PREDICTION

by

PATRA VOLARATH

Committee Chair:	Robert Harrison
Committee Members:	Irene Weber
	Pan Yi

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

May 2008

ACKNOWLEDGEMENTS

I would like to, first of all, give a great gratitude to all my parents, both here in the United States and in Thailand, and my belated grandmother for their patience during my so called "option-exploring" years. Next, I would like to give a gratitude to my advisor, Dr. Robert Harrison, for providing me an opportunity to work in this project. Understanding how computer technology and mathematical equations could be used to study the biological systems has always been a dream of mine, and entering the field from a biology-based training was a great challenge. I am grateful for his patience and understanding during my years under his guidance. I would also like to thank Dr. Irene Weber and her lab members, particularly Yuan-Fang Wang and Dr. Johnny Agniswamy, for their advice on my writing and presentation skills. I would also like to acknowledge my friends: Brandi Brooks, Patricia Springer, Jeanetta Holley, Laura Copp, Chuck Woods, Miki Kassai, Gaurav Arora, and Tracie Tie. Some of them have been with me since the time I joined GSU (and even before). All of them have always been there when I needed someone to lean on. I would like to thank my friends from the computer sciences department: Gulsah Altun, Hae-Jin Hu, and Hao Wang. They have been great in helping me comprehending the complicated subjects in computer sciences. And last, but not least, I would also like to thank my friends Judy Block and everyone from the GSU Student Support Services, particularly Deidre Steed and Martha Fowler, for giving me an opportunity to explore my teaching capability and meet great people.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iv

LIST OF TABLES x

LIST OF FIGURES xi

CHAPTER

1. Background Information of Computer-Aided Organic Synthesis Programs 1

2. Compound Representations, Molecular Modeling, and Energy Minimization 8

2.1 Compound Representations 8

2.1.1 Molecular Graphs and Trees 8

2.1.2 Matrix Representation 10

2.1.3 Connection Table 11

2.1.4 Line Notations 12

2.2 Molecular Modeling 14

2.2.1 Formulation of Force Fields 15

A. Bonding Potentials 16

Stretching Energy 16

Bending Energy 17

Torsion Energy 17

Improper Torsions and Out-of-Plane Bending Motions 18

Pyramid Height Terms 19

B. Nonbonding Potentials 20

Electrostatic Potential 20

Dipole-Dipole Interactions 21

Van der Waals Interactions 22

Hydrogen Bonds 25

Cross Terms 26

2.2.2 Atom Types and Paramarization 28

2.2.3 Currently Available Force Fields 30

A. The Allinger MM Force Fields 30

B. AMBER Force Field 32

C. CHARMM Force Field 33

D. UFF Force Field 34

2.2.4 Force Fields Performances 34

2.2.5 Choosing Force Fields 38

2.3 Energy Minimization 39

2.3.1 Derivatives 42

A. First-Order Minimization Methods 43

Steepest Descent Algorithm 44

Conjugate Gradient Algorithm 44

B. Second Derivative Methods 45

Newton-Raphson Algorithm 45

Block-Diagonal Newton-Raphson Algorithm 46

Quasi-Newton Methods 46

2.3.2 Heuristic Methods for Global Minima Searching 47

A. Simulated Annealing 47

B. Genetic Algorithm 48

C. Molecular Dynamics 49

D. Penalty Methods 50

3. Knowledge-based algorithms for chemical structure and property analysis 56

3.1 Methodology 56

3.2 Results and Discussion 57

3.2.1 Structural-Based Chemistry Determination 57

3.2.2 Structural Search Algorithm 60

A. Choosing a unique root and 'best-tree' algorithm 60

B. Building a molecular tree 61

C. Exact- and Sub- Structure Searching 62

3.3 Conclusion 62

4. Program Designs for Chemical Functional Groups and their Reactivity 65

4.1 Program Designs 67

4.1.1 Functional Group Search Programs (FGSP) 67

4.1.2 Functional Group Scoring Program 68

A. The Atom Sharing Impact (ASI) on the TFG's Overall Reactivity 68

B. Affects of Neighboring Atoms on the OTFGR 71

4.2 Results 72

4.2.1 Analysis of the Functional Group Search Programs 72

4.2.2 Analysis of the Functional Group Scoring Function 77

4.3 Conclusion 81

5. Term-Rewriting Grammar for Predicting Basic Chemical Reactions 83

5.1 Chemical Term and Reaction Designs 83

5.1.1 Maude Programming for Chemical Reactions 83

5.1.2 Designing of Terms 86

5.1.3 Designing of Reaction Rules 89

5.2 Structure Handling Programs 91

5.2.1 Functional Group Pattern Generation Programs (FGPGPs) 92

5.2.2 Product Modeling Programs (PMPs) 93

5.2.3 Modeling of Products of Two Compounds 94

5.3 Results 95

5.3.1 Basic Functional Group Reactions (BFGRs) 95

5.3.2 Advanced Chemical Reactions 99

A. Mechanism-Emphasis Reactions: Substitutions and Eliminations Mechanisms 99

5.4 Discussion 109

6. Application of Term-Rewriting Grammar in Describing Complicated Reactions 111

6.1 Acetoacetic Ester Synthesis 111

6.2 Alder Ene Reaction 116

6.3 Gassman Indole Synthesis 120

6.4 Discussion 125

7. Project Conclusion 127

8. APPENDICES 128

Appendix A: List of the Functional Group Search Algorithm Tables 129

Appendix B: Electronegativity Chart 135

Appendix C: Detail coding in Ester.maude 136

Appendix D: Detailed Coding in Acid.maude 138

LIST OF TABLES

Table 2.1: SMILES notations for nine compounds 14

Table 5.1: The three modules in the Alcohol.maude program and their rules 92

LIST OF FIGURES

Figure 1.1:	Retrosynthesis Scheme. The structure on the left is the target structure and the structures on the right are the potential starting materials chemoinformatics	2
Figure 1.2:	Retrosynthesis tree with formal method	3
Figure 1.3:	A Forward Search Scheme. The different colors in the triangles indicate that the two are similar but not exactly matched	3
Figure 2.1:	Common representations of chemical rings	9
Figure 2.2:	A graph and a tree representation of benzoic acid. In the tree, the cyclic structure is indicated by the duplication of the atom number 3	10
Figure 2.3:	A matrix representation of benzoic acid. The numbers 1-9 in the first column and the first row represent the atoms in the molecule. Inside each of the cells, 1 indicates bonding between the atom pair, and 0 indicate otherwise	11
Figure 2.4:	Connectivity table of aspirin generated by ChemDraw Ultra 6.0	12
Figure 2.5	Typical torsion angle of bonded atoms	18
Figure 2.6	Relative Performances of Different Force Fields. The relative performance of different potential sets is shown in this figure which shows the RMS error (in kcal/mol) for several sets on a peptide benchmark	30
Figure 2.7	Typical Energy Minimization Scheme	40
Figure 2.8	The modified scheme for finding energy minima on the potential energy surface	41
Figure 4.1:	Examples of commonly found functional groups in organic chemistry	65
Figure 4.2:	Molecules containing several functional groups	66
Figure 4.3:	The thirteen test functional groups for this work	67
Figure 4.4:	The signature pattern of the amide are enclosed in red	68
Figure 4.5:	The amide atoms are enclosed in the blue box, and the amine atoms are enclosed in the red box	70
Figure 4.6:	Two representations of benzene rings	73

Figure 4.7:	Different representations of benzene rings	73
Figure 4.8:	Linear ring representations that are recognizable by the program	74
Figure 4.9:	The non-linear ring forms that are recognizable by the algorithm. In the last structure, only the rings in blue are recognizable by the program	74
Figure 4.10:	Test structure for branched rings	74
Figure 4.11:	The structure of the test molecule, Molecule 1	75
Figure 4.12:	Combined output of test Molecule 1	76
Figure 4.13:	Result from the scoring program for Molecule 1	78
Figure 4.14:	The structure of a test molecule (Molecule 2) that contains a symmetric functional group	79
Figure 4.15:	Reactivity result for Molecule 2	80
Figure 5.1:	The programming in the acid.maude	85
Figure 5.2:	An example output from Maude when a .maude is loaded	86
Figure 5.3:	A nucleophilic reaction mechanism between an amine and a carbonyl compound	87
Figure 5.4:	Baeyer-Drewson Indigo Synthesis	87
Figure 5.5:	An example molecule that shows how the atomtypes can be used to label the atoms in the molecule	88
Figure 5.6:	An overall reaction pathways of an ester	99
Figure 5.7:	The descriptions of the primary, secondary, and tertiary carbons	100
Figure 5.8:	The back-side attack in the Sn2 reaction	100
Figure 5.9:	The steric hindrance of the tertiary carbon prevents the nucleophile from attacking the carbon	100
Figure 5.10:	The mechanism of the nucleophilic reaction with the tertiary carbon	101
Figure 5.11:	The pathways of the SN1 and E1 reactions	102

Figure 5.12:	The E2 mechanism	102
Figure 5.13:	Nucleophilic reaction between the amine and the 2-chloro-butane	105
Figure 5.14:	The substitution (SN1) reaction of the methanol and 2-chloro-tert-butane	106
Figure 5.15:	2-chloro-ethylcyclohexane	107
Figure 5.16:	Hydride shifting between the clorine (Cl) and the hydride (blue H)	108
Figure 6.1:	Acetoacetic ester synthesis	111
Figure 6.2:	Atom labeling of the acetoacetic ester synthesis starting compound	111
Figure 6.3	The two modules in AcetoAceticEsterSynthesis.maude. The lines enclosed in the red boxes are the rules describing the two steps in the reaction	113
Figure 6.4:	Term-rewriting result from MODULE1 for input (C3-1, C2-2, C3-3, C3-4, O3-5, C3-6, C3-7). The last line is the returned result	114
Figure 6.5:	Output from MODULE 2 for (C3-1, C2-2, anion-C3-3, C3-4, O3-5, C3-6, C3-7)	114
Figure 6.6:	The second–half reaction for acetoacetic ester synthesis	115
Figure 6.7:	The reaction pathway for the intermediate product through the dilute acid or alkali (MODULE3). The numbers in red show the decomposition of the atoms as a result of the reaction, and the information enclosed in the red box is the result returned by the program	115
Figure 6.8:	The reaction pathway for the intermediate product through the concentrated alcoholic alkali (MODULE4). The numbers in blue show the decomposition of the atoms as a result of the reaction, and the information enclosed in the blue box is the result returned by the program	116
Figure 6.9:	Outline reaction for alder ene. The focus of the reaction is enclosed in the blue box	117
Figure 6.10:	Two examples of Alder ene reactions. The reactive portions of the reactants are enclosed in the blue and red boxes	117
Figure 6.11:	Transformation rules describing the Alder ene reaction. The rules are declared separately in three different modules. The color of the	118

	borderline surrounding each module matches the color of the borderline surrounding the fragment that the module describes	
Figure 6.12:	Results obtained the three modules of Alder ene reaction. The result from each module is indicated by a colored arrow. These arrows are color coded to match their fragments	120
Figure 6.13:	Gassman Indole Synthesis	121
Figure 6.14:	The rules describing the transformation of the terms describing Structure A to those describe Structure B. The terms printed in blue represent those atoms that are included in the blue circle	122
Figure 6.15:	Rules describing the term transformation from Structure B to Structure C. The blue terms in line 5 are those atoms that are involved in the indole structure in Structure C	123
Figure 6.16:	Rules describing the term transformation from :indole to a cation indole (+indole)	124
Figure 6.17:	Rules describing term transformation from +indole and the neutral indole	125
Figure 7.1:	Overall design and applications of the algorithms (blue) developed in this project	127

Chapter 1: Background Information of Computer-Aided Organic Synthesis Programs

Organic synthesis is significant not only because that it is a step-by-step manual that informs chemists of how to synthesize compounds, but also its knowledge can be used to suggest synthetic pathways of novel compounds that have never been synthesized before. The origin of organic synthesis can be dated back to the early 1800s. Since then, numerous reactions have been discovered and published in the literatures. In 1960's, computer-aided organic synthesis (CAOS) planning programs were introduced. [1, 2]. LHASA (Logic and Heuristics Applied to Synthetic Analysis), for example, was one of the pioneer CAOS programs developed by E. J. Corey at Harvard University to assist the chemists with their synthesis planning. [3-6] Since LHASA, other CAOS programs, which have similar fundamental designs to the LHASA but with additional features that enhance their prediction, have been developed. Simulation and Evaluation of Chemical Synthesis (SECS) is a derivative program of LHASA. [7-9] One of the crucial features of SECS is that the program uses the stereo-chemical features of the target molecule to screen for suitable structure transformations. The program also understands the 3D structure of the target. This allows the program, for example, to perform an energy minimization that can assist in the synthetic pathway determinations. SYNCHEM has a design that is also similar to the LHASA. [2] Its effort is to apply artificial intelligence and heuristic programming into the synthetic pathway predictions.

The synthesis prediction approaches can be divided into three broad categories: retrosynthesis [10-12], formal approach [13-18], and forward approach [9, 19-25]. The retrosynthesis is a synthesis strategy that reversely synthesizes a target compound to their starting materials, Figure 1.1.

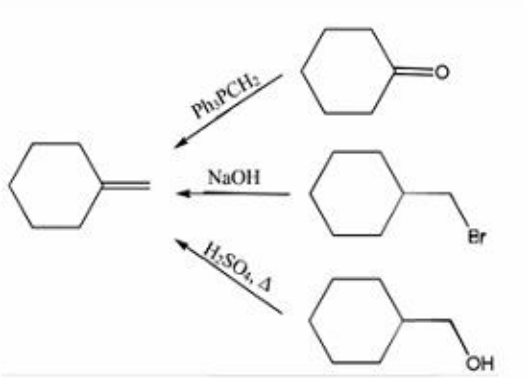


Figure 1.1: Retrosynthesis Scheme. The structure on the left is the target structure and the structures on the right are the potential starting materials [26]

The structure transformation rules in the retrosynthesis, or the synthon approach [27-31], are heuristic and developed based on the programmer's perception of the chemical reactions. The CAOS programs mentioned, thus far, use the retrosynthesis as their main or a part of their main prediction strategy. The size of the retrosynthesis tree depends on the complexity of the starting compound. The larger and the more complex is the starting compound, the larger is its retrosynthesis tree. As a result, this method often suffers from the combinatorial explosion of the retrosynthesis tree. In addition, the transformation knowledge is a perception-based, which is incomplete. For this reason, the programs often predicted unsynthesizable intermediates and products, and cause some of the 'real' compounds to be missed. To resolve this problem, a method called a formal method is incorporated into the retrosynthesis to improve the prediction results. [32]

The formal method does not require any knowledge of known or good reactions. The method can draw upon all theoretical transformations, even those that do not have supportive experimental data. One advantage provided by the formal method is that it can suggest synthetic pathways that might have been overlooked by the chemists. However, since there are no

restrictions to evaluate the results, the combinatorial problem with the formal method is more severe than that of the retrosynthesis alone, Figure 1.2.

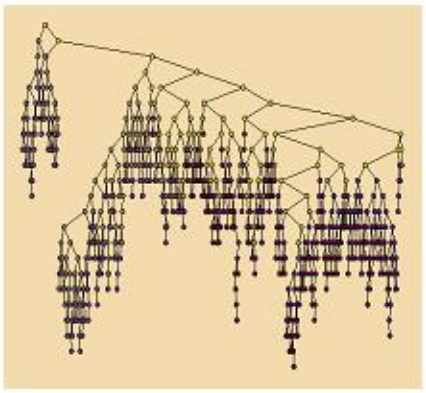


Figure 1.2: Retrosynthesis tree with formal method (http://cg.scs.carleton.ca/~luc/trees.html)

As shown in Figure 1.2, the results from the formal method are numerous. It is, therefore, impractical for the chemists to validate all of the predictions. Some of the prediction programs the employ that formal approach include IGOR (Interactive Generation of Organic Reactions) [13], EROS [15, 33], and SYNGEN [16, 17, 34].

To alleviate the combinatorial problem, another approach called a forward search is introduced to work collaboratively with the retrosynthesis.

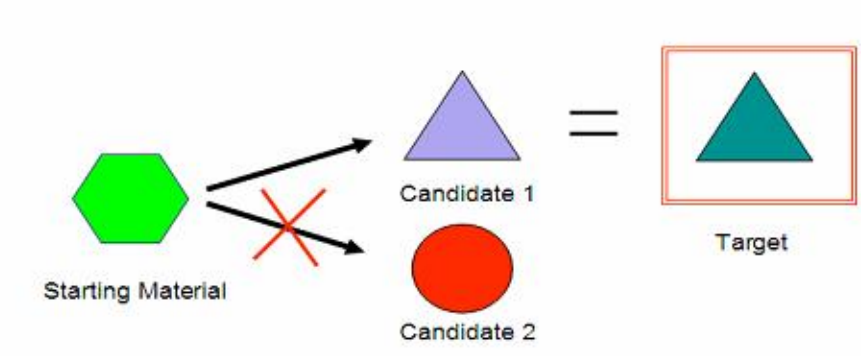


Figure 1.3: A Forward Search Scheme. The different colors in the triangles indicate that the two are similar but not exactly matched

This method requires the knowledge of the starting material and the target molecules. In Figure 1.3, for each starting material (the hexagon), molecular fragments in a database are searched for a set of potential candidates (the purple triangle and the red circle) that share characteristics with the target (the green triangle). The characteristics between the target and each of the candidates are compared, and only the candidates whose characteristics are highly similar to the target are selected. The comparison is usually carried out through several algorithms that have been developed to fine-tune the comparison quality. The process may be lengthy depends on the size and the number of the samples that are to be compared. This may be problematic, particularly, if a large number of the compounds are to be processed simultaneously and continuously. The CHIRON [20, 35, 36] program is designed to search for candidates with a maximum criteria match (carbon skeleton, functionality, and stereochemistry) to the target compound. The candidates are searched in the commercially available databases. CAMEO (Computer Assisted Mechanistic Evaluation of Organic Reactions), another program, is designed to understand (from the programmer's perceptions) the reaction pKa values, as well as recognize the nucleophiles and electrophiles. [37, 38] The output obtained from the program is then screened to determine the stable compounds. CAMEO has been applied to mechanistic studies of a variety of chemical reactions. [26, 39, 40]

Thus far, these programs have been successfully applied to only some synthetic pathway predictions. The major sources of their limitations are the incomplete knowledge base and the complicated designs in their prediction systems. The goal of this dissertation is to design a more efficient prediction approach of the chemical reactions for small organic molecules. We are interesting in exploring a term-rewriting grammar, which is a formal grammar that describes transformations of terms. The transformation can be logically represented as followed:

$$A + B \Rightarrow C$$

where 'A' and 'B' are input terms and 'C' is a transformed term

The transformation above is performed based on a set of rules. This operation closely resembles the way organic syntheses are planned in the lab. Hence, we are interested in determining if the term-rewriting approach is capable of carrying out such a prediction task. In the following chapters, the basics of the cheminformatics related to this work are first discussed (Chapter 2), followed by the design of the 2D structure interpretation algorithms (chapters 3 and 4) and the design of the term-rewriting rules for the chemical reactions (Chapter 5). The paper is, then, concluded by the final chapter (Chapter 6) that demonstrates how the developed term rewriting approach can be applied to describe the advanced reactions and how developed algorithms are put together to form a chemical reaction prediction system for small molecules.

REFERENCES

1. Barone, R.; Chanon, M., In *Encyclopedia of Computational Chemistry*, Schleyer, P. V. R., Ed. Wiley: Chichester, 1998; pp 2931–2948.
2. Zefirov, N. S.; Gordeeva, E. V., Computer-assisted Synthesis. *Russ. Chem. Rev.* **1987,** 56, 1002-1014.
3. Johnson, A. P.; Marshall, C., Starting material oriented retrosynthetic analysis in the LHASA program. 3. Heuristic estimation of synthetic proximity. **1992,** 32, (5), 426-429.
4. Johnson, A. P.; Marshall, C., Starting material oriented retrosynthetic analysis in the LHASA program. 2. Mapping the SM and target structures. **1992,** 32, (5), 418-425.
5. Mata, P.; Lobo, A. M.; Marshall, C.; Johnson, A. P., Implementation of the Cahn-Ingold-Prelog System for Stereochemical Perception in the LHASA Program. **1994,** 34, (3), 491-504.
6. Ott, M. A.; Noordik, J. H., Long-Range Strategies in the LHASA Program: The Quinone Diels-Alder Transform. **1997,** 37, (1), 98-108.
7. Wipke, W. T.; Dyott, T. M., Simulation and evaluation of chemical synthesis. Computer representation and manipulation of stereochemistry. **1974,** 96, (15), 4825-4834.
8. Wipke, W. T.; Gund, P., Simulation and evaluation of chemical synthesis. Congestion: a conformation-dependent function of steric environment at a reaction center. Application with torsional terms to stereoselectivity of nucleophilic additions to ketones. **1976,** 98, (25), 8107-8118.
9. Wipke, W. T.; Rogers, D., Artificial intelligence in organic synthesis. SST: starting material selection strategies. An application of superstructure search. *J. Chem. Inf. Comput. Sci.* **1984,** 24, 71–81.
10. Gehrke, N.; Nassif, N.; Pinna, N.; Antonietti, M.; Gupta, H. S.; Colfen, H., Retrosynthesis of Nacre via Amorphous Precursor Particles. **2005,** 17, (26), 6514-6516.

11. Nicolaou, K. C.; Nantermet, P. G.; Ueno, H.; Guy, R. K.; Couladouros, E. A.; Sorensen, E. J., Total Synthesis of Taxol. 1. Retrosynthesis, Degradation, and Reconstitution. **1995,** 117, (2), 624-633.
12. Paquette, L. A.; Zhao, M., Enantiospecific Total Synthesis of Natural (+)-Taxusin. 1. Retrosynthesis, Advancement to Diastereomeric trans-Δ9,10-Tricyclic Olefinic Intermediates, and the Stereocontrol Attainable Because of Intrinsic Rotational Barriers Therein. **1998,** 120, (21), 5203-5212.
13. Ugi, I.; Bauer, J.; Bley, K.; Dengler, A.; Dietz, A.; Fontain, E.; Gruber, B.; Herges, R.; Knauer, M.; Reitsam, K.; Stein, N., Computer-Assisted Solution of Chemical Problems - The Historical Development and the Present State of the Art of a New Discipline of Chemistry. *Angew. Chem., Int. Ed. Engl.* **1993,** 32, 201–227.
14. Dengler, A.; Fontain, E.; Knauer, M.; Stein, N.; Ugi, I., Competing concepts in CAOS. *Recl. Trav. Chim. Pays-Bas* **1992,** 111, 262–269.
15. Gasteiger, J.; Jochum, C., EROS, a computer program for generating sequences of reactions. *Top. Curr. Chem.* **1978,** 74, 93–126.
16. Hendrickson, J. B., Organic Synthesis in the Age of Computers. *Angew. Chem., Int. Ed. Engl.* **1990,** 29, 1286–1295.
17. Hendrickson, J. B.; Grier, D. L.; Toczko, A. G., A logic-based program for synthesis design. *J. Am. Chem. Soc.* **1985,** 107, 5228–5238.
18. Herges, R.; Hoock, C., Reaction planning: computer-aided discovery of a novel elimination reaction. *Science* **1992,** 255, 711–713.
19. Mehta, G.; Barone, R.; Chanon, M., Computer-Aided Organic Synthesis - SESAM: A Simple Program to Unravel Hidden Restructured Starting Materials Skeleta in Complex Targets. *Eur. J. Org. Chem.* **1998,** 1998, (7), 1409–1412.
20. Hanessian, S.; Franco, J.; Larouche, B., The psychobiological basis of heuristic synthesis planning - man, machine, and the chiron approach. *Pure Appl. Chem.* **1990,** 62, 1887–1910.
21. Salatin, T. D.; Jorgensen, W. L., Computer-assisted mechanistic evaluation of organic reactions. 1. Overview. *J. Org. Chem.* **1980,** 45, 2043–2051.
22. Fleischer, J. M.; Gushurst, A. J.; Jorgensen, W. L., Computer Assisted Mechanistic Evaluations of Organic Reactions. 26. Diastereoselective Additions: Cram's Rule. *J. Org. Chem.* **1995,** 60, 490–498.
23. Ihlenfeldt, W.-D.; Gasteiger, J., Computer-assisted planning of organic syntheses: the second generation of programs. *Angew. Chem., Int. Ed. Engl.* **1995,** 34, 2613–2633.
24. Fick, R.; Ihlenfeldt, W.-D.; Gasteiger, J., Computer-assisted design of synthesis for heterocyclic compounds. *Heterocycles* **1995,** 40, 993–1007.
25. Braban, M.; Pop, I.; Willard, X.; Horvath, D., Reactivity Prediction Models Applied to the Selection of Novel Candidate Building Blocks for High-Throughput Organic Synthesis of Combinatorial Libraries. *J. Chem. Inf. Comput. Sci.* **1999,** 39, 1119–1127.
26. Helson, H. E.; Jorgensen, W. L., Computer-Assisted Mechanistic Evaluation of Organic Reactions. 25. Structure Diagram Positioning. **1994,** 34, (4), 962-971.
27. Cohen, N.; Eichel, W. F.; Lopresti, R. J.; Neukom, C.; Saucy, G., Synthetic studies on (2R,4'R,8'R)-alpha-tocopherol. An alternative synthesis of (2R,6R)-(+)-2,6,10-trimethylundecan-1-ol, a key side chain synthon. *J Org Chem* **1976,** 41, (22), 3512-5.
28. Lynch, V. M.; Li, W.; Martin, S. F., Structure of the ABC ring synthon of dendrobine. *Acta Crystallogr C* **1988,** 44 (Pt 1), 187-9.

29. Matyska, L.; Koca, J., MAPOS: a computer program for organic synthesis design based on synthon model of organic chemistry. *J Chem Inf Comput Sci* **1991,** 31, (3), 380-6.
30. Shibasaki, H.; Furuta, T.; Kasuya, Y., Stereochemical aspects in the enantioselective synthesis of cortisone by the indan synthon method. *Steroids* **1992,** 57, (7), 325-7.
31. Konno, K.; Maki, S.; Fujishima, T.; Liu, Z.; Miura, D.; Chokki, M.; Takayama, H., A novel and practical route to A-ring enyne synthon for 1 alpha,25-dihydroxyvitamin D3 analogs: synthesis of A-ring diastereomers of 1 alpha,25-dihydroxyvitamin D2 and 3-methyl-1,25-dihydroxyvitamin D3. *Bioorg Med Chem Lett* **1998,** 8, (2), 151-6.
32. Todd, M. H., Computer-Aided Organic Synthesis. *Chem. Soc. Rev.* **2005,** 34, 247-266.
33. Gasteiger, J.; Herwig, A., Simulation of Organic Reactions: From the Degradation of Chemicals to Combinatorial Synthesis. *J. Chem. Inf. Comp. Sci.* **2000,** 40, 482-494.
34. Hendrickson, J. B.; Toczko, A. G., SYNGEN program for synthesis design: basic computing techniques. **1989,** 29, (3), 137-145.
35. Hanessian, S.; Botta, M.; Larouche, B.; Boyaroglu, A., Computer-assisted perception of similarity using the Chiron program: a powerful tool for the analysis and prediction of biogenetic patterns. **1992,** 32, (6), 718-722.
36. Hanessian, S.; Franco, J.; Gagnon, G.; Laramee, D.; Larouche, B., Computer-assisted analysis and perception of stereochemical features in organic molecules using the CHIRON program. **1990,** 30, (4), 413-425.
37. Gushurst, A. J.; Jorgensen, W. L., Computer-assisted mechanistic evaluation of organic reactions. 12. pKa predictions for organic compounds in Me2SO. **1986,** 51, (18), 3513-3522.
38. Bures, M. G.; Roos-Kozel, B. L.; Jorgensen, W. L., Computer-assisted mechanistic evaluation of organic reactions. 11. Electrophilic aromatic substitution. **1985,** 50, (23), 4490-4498.
39. Roos-Kozel, B. L.; Jorgensen, W. L., Computer-assisted mechanistic evaluation of organic reactions. 2. Perception of rings, aromaticity, and tautomers. **1981,** 21, (2), 101-111.
40. Peishoff, C. E.; Jorgensen, W. L., Computer-assisted mechanistic evaluation of organic reactions. 4. Organosilicon chemistry. **1983,** 48, (12), 1970-1979.

Chapter 2: Compound Representations, Molecular Modeling, and Energy Minimization

Cheminformatics is a branch of computational chemistry that covers a wide range of topics regarding the representations, the handling, and the interpretations of chemical information. This work deals with interpretations of a 2D representation of molecules, as well as with molecular modeling using AMMP (Advanced Molecular Modeling Program) as the primary modeling tool. It is, therefore, useful to discuss the different types of compound representations in the computer, as well as the commonly used techniques for molecular modeling and energy minimization.

2.1 Compound Representations

2.1.1 Molecular Graphs and Trees

Molecules can be expressed in a form of a graph or a tree, which consists of a series of nodes (representing the atoms) that are connected by edges (representing the bonds). This is one of the preferred methods of representing the molecules because, first of all, it mimics the way atoms are bond in the real molecules, and, second of all, the structural information and properties can be derived using operations of graph theory.[1-3]

Graph representation is a most preferred representation of molecules. In practice, several issues must be considered when converting the molecule into a graph. An example is the representations of the ring system. Aspirin, for instance, can have three different graphs that are chemically equivalent. One way of representing an aromaticity within a ring is to define the bond value of aromatic system to be a specially fixed value. Advanced Molecular Model Program (AMMP) defined this value as 1.5. This rule is enforced in all the inputs and data in the chemical database to ensure structural compatibility during structure comparison, Figure 2.1.

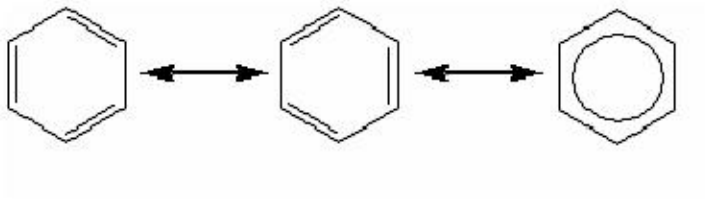


Figure 2.1: Common representations of chemical rings

The difference between the graph and the tree representations is that cyclic structures are not allowed in the tree, Figure 2.2. Instead, a cyclic structure is indicated by a duplication of a joined atom. The advantage of using the tree representation is that the tree comparison can be completed within polynomial time; this makes the tree more favorable than the graph, whose problems are usually NP problems. The breakage of the cyclic structure in the tree is determined by a set of subjective rules. The mapping of a molecule to tree is not unique. In substructure comparison, the tree representation may need to be rebuilt to compensate for the possible loss of the connectivity information caused by the cycle breakage. However, the tree representation greatly reduces the computation complexity imposed on the graph problems. The choice of which of the representations to use depends on the program users.

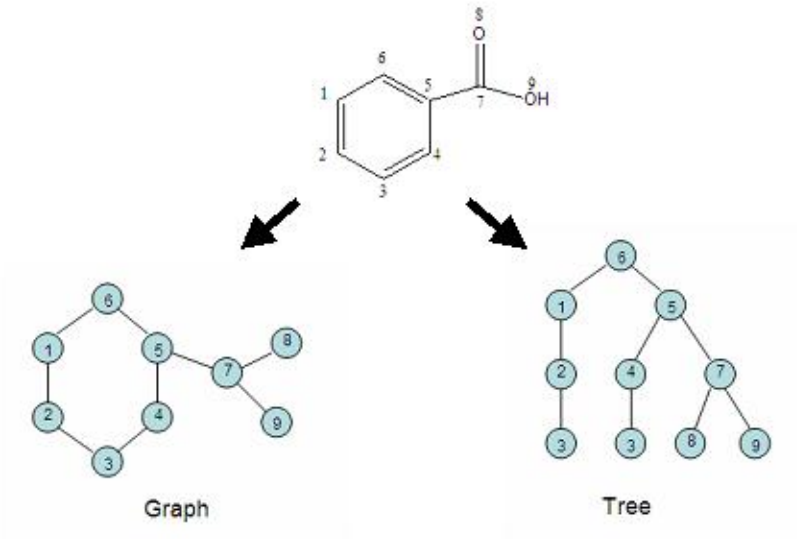
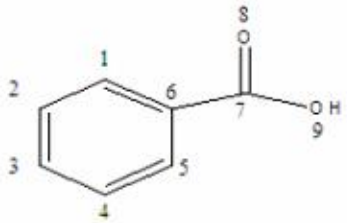


Figure 2.2: A graph and a tree representation of benzoic acid. In the tree, the cyclic structure is indicated by the duplication of the atom number 3

2.1.2 Matrix Representation

Molecular graphs can also be represented as matrices, Figure 2.3. An advantage is that the calculation of paths and cycles (for ring structures) can be performed quickly by matrix operations. A typical molecular matrix consists of *n* rows and *n* columns (*n* x *n* matrix) for a molecule with *n* atoms. The matrix elements can be any data (a bond order, a physical distance, etc.) that represents a relationship between a given pair of atoms.[4] This depends on the type of information that is being emphasized. Some of matrices that have been used in chemoinformatics are adjacency, distance, incidence, bond, and bond-electron matrices.[5]

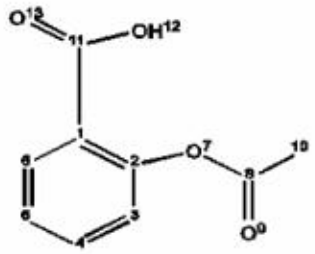


	1	2	3	4	5	6	7	8	9
1	0	1	0	0	0	1	0	0	0
2	1	0	1	0	0	0	0	0	0
3	0	1	0	1	0	0	0	0	0
4	0	0	1	0	1	0	0	0	0
5	0	0	0	1	0	1	0	0	0
6	1	0	0	0	1	0	1	0	0
7	0	0	0	0	0	1	0	1	1
8	0	0	0	0	0	0	1	0	0
9	0	0	0	0	0	0	1	0	0

Figure 2.3: A matrix representation of benzoic acid. The numbers 1-9 in the first column and the first row represent the atoms in the molecule. Inside each of the cells, 1 indicates bonding between the atom pair, and 0 indicate otherwise

2.1.3 Connection table

A connection table for a molecule consists of two sections: a list of atoms that are in a molecule and a list of their bond connectivity.[6, 7] A MDL .mol file is a well-known file format that uses the connection table to represent the molecules, and a typical MDL .mol file is shown in Figure 2.4. The first section contains description of the atoms, and these include the Cartesian coordinates, the atomic symbol, followed by columns of property descriptions (e.g., atomic charges, hybridization state). The chemistry of molecule is more completed in the connectivity table than in the graph/tree form because the properties (usually represented via numerical values) can be expressed in the columns. In MDL .mol file, hydrogen atoms are usually omitted in the 2D .mol file, but are included in the 3D version.




```
aspirin.mol
 ChemDraw08140618222D
13 13  0  0  0  0  0  0  0  0999 V2000
  -1.1953    0.2062    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -0.4809   -0.2062    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -0.4809   -1.0313    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.1953   -1.4438    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.9098   -1.0313    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.9098   -0.2062    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   0.3160    0.0073    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   1.1129   -0.2062    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
   1.1129   -1.0313    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
   1.9098    0.0073    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -1.1953    1.0313    0.0000 C   0  0  0  0  0  0  0  0  0  0  0  0
  -0.4809    1.4438    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
  -1.9098    1.4438    0.0000 O   0  0  0  0  0  0  0  0  0  0  0  0
 1  2  2  0
 2  3  1  0
 3  4  2  0
 4  5  1  0
 5  6  2  0
 6  1  1  0
 2  7  1  0
 7  8  1  0
 8  9  2  0
 8 10  1  0
 1 11  1  0
11 12  1  0
11 13  2  0
M  END
```

Figure 2.4: Connectivity table of aspirin generated by ChemDraw Ultra 6.0

2.1.4 Line Notations

Another way of expressing molecules is through the use of a linear or string notation. A linear notation uses characters and digits to encode the molecular structure. Linear notations are more compact than the graph/tree and the connection table forms. Therefore, linear notations are very useful for storing and transmitting large numbers of molecules, particularly when used in conjunction with relational databases that support only characters and digits. Linear notations

can be rapidly searched for approximate similarities using string search algorithms similar to those used for sequence alignment such as BLAST [8-10], but the results can be relatively inaccurate.

An early line notation that became widely used was the Wiswesser Line Notation (WLN).[11] WLN uses a complex set of rules to represent different functional groups and their connectivity in molecules. A more recent linear notation that has been developed and became widely accepted is the SMILES notation.[12, 13]

In SMILES, the atoms are represented by the atomic symbols. The elements with the number of attached hydrogens conform to the lowest normal valence consistent with explicit bonds can be written without brackets.[14] For example, water molecule can be expressed as O; while, hydronium cation is expressed as [OH3+]. No information on three-dimensional arrangement of atoms is represented, and hydrogen atoms can be omitted or included. Single, double, triple and aromatic bonds are represented by the symbols -, =, #, and:, respectively, and branches are specified by parentheses. Atoms in aromatic rings are represented by lower case letters, and the aromatic bonds are usually omitted. Cyclic structures are represented by breaking one single (or aromatic) bond in each ring. The bonds are numbered in any order, designating ring-opening (or ring-closure) bonds by a digit immediately following the atomic symbol at each ring closure. Pyridine, for instance, can be expressed in SMILES as n1ccccc1. Table 2.1 shows SMILES for some of the organic compounds.

Table 2.1: SMILES notations for nine compounds [14]

Compound Names	SMILES Notations
Butane	CCCC
Isobutene	C(C)(CCl)C
Anthracene	C(c(ccc1)ccc2)(c1)c2
Benzene	c1ccccc1
Naphthalene	c1ccc2ccccc2c1
Methylcyclohexane	CC1CCCCC1
Trichloromethane	C(CL)(CL)CL
Nitromethane	CN(=O)=O
1,3-cyclohexadiene	C1=CC=CCC1

Since the publication of its original version, SMILES has been improved and made into different versions. Canonical SMILES is a version of the SMILES that was designed to ensure a single unique SMILES representation for a chemical compound. Another version of SMILES called Isomeric SMILES extends the string representation to support the chemical properties such as isotopes, chirality, and configuration about double bonds.[14] Other descendents of SMILES include SMARTS [15] that describes molecular patterns, which is useful for structure comparison, and SMIRKS [16] that contains rules for molecular transformations, which is useful for chemical reactions.

2.2 Molecular Modeling

Structure prediction remains one of the most challenging tasks in modern molecular research. A number of major research areas such as drug design and structural genomics spend a significant amount of time on structural analysis, which is a very time-consuming process. The main goal of structure prediction is to shorten the amount of time spending in this process, while maintaining the accuracy of the prediction. If this goal is accomplished successfully, it can significantly facilitate the productivity of the future research. However, molecular behaviors are

complicated and not yet fully understood. Predicting behaviors of protein structure in solutions, for example, involves intensive computational analysis of both the protein and the solvent.

One way of handling such a problem in molecular simulation is the use of potential equations that describe molecular behaviors. This method is sometimes referred to as *potential-based simulation* method, or *force field* method. Force fields are significantly useful to structural simulation because they contain information that can be used to derive molecular properties. These include thermodynamic properties, molecular energies, molecular motions in space, and time-dependent conformations of molecules. However, most of the currently available force fields can only successfully predict the structures in certain situations, but not in others.

Because of this limitation, new force fields are constantly being developed. Some have been designed from scratch; while, others are modifications of their previous versions. A goal is to improve the accuracy of the predictions, as well as to establish a force field that can be applied to all biological systems.

The goal of this review is to give a basic insight into the current knowledge of the potential-based simulation method, as well as its role in the modern molecular modeling.

2.2.1 Formulation of Force Fields

Force fields are generally the sums of all energy terms that describe the structural properties of molecules, which can be grouped into two categories: bonding and nonbonding. These include bonds, angles, torsions, vdw (van derWaals), and electrostatic interactions. A general format of force fields can be written as an abstract of the bonding and non-bonding potentials (V), as shown in Equation 3.1 or in terms of individual energy component (E), as shown in Equation 3.2:

$$V_{\text{total}} = \sum (V_{\text{bonding}} + V_{\text{nonbonding}}) \qquad (2.1)$$

$$V(\text{conformations}) = E_{\text{bonds}} + E_{\text{angles}} + E_{\text{torsion}} + E_{\text{vdw}} + E_{\text{electrostatic}} + E_{\text{other}} \qquad (2.2)$$

The bonding potentials include the physical descriptions of chemical bonds in the molecule: bond stretching and/or compressing (bond length), bond bending, and torsion angles. On the other hand, the nonbonding potentials comprise of interaction energy in the molecule: electrostatic interaction, dipole-dipole interactions, van derWaals interactions, and hydrogen bonds. Additional terms such as out-of-plane angle bending, improper torsions interaction, and cross terms (bond-stretch-angle-bend, angle-bend-angle-bend, etc.) are also often founded in force fields. The significance of these additional terms is to describe the potential within molecules in greater details.

When designing a new force field, it is imperative to consider both the conservation of energy and angular momentum such that the sum of all forces equals to zero. Otherwise, the behaviors of the simulating molecules can be unpredictable. An example is a molecule moves or spins on its own without any applied forces. So far, the potential terms within the currently available force fields have been carefully designed to satisfy this requirement. However, it is still important to verify any possible numerical and/or programming errors that possibly result in a physical behavior.

A. Bonding potentials

The bonding energy is modeled in a force field as a distance-dependent function. An equilibrium distance r_0 is defined for the standard length of the chemical bond, where no force is exerted on either of the bonded atoms. For a distance that $r \neq r_0$, the atoms are forced towards r_0 by the potential energy surface. In most force fields, the energies that contribute to the bonding potential are: stretching, bending, and torsion energies.

Stretching Energy

In molecular mechanics, the chemical bond is treated as a spring that has an equilibrium length. Stretching or compressing the spring from r_0 requires an applied force. Therefore, the

potential energy of a chemical bond V_{bond} depends on the deformation of the spring from r_0, and on the spring constant k_{bond}.

$$V_{bond} = k_{bond}(r-r_0)^2 \qquad (2.3)$$

The treatment of the chemical bond as a spring is approximated. Since the potential, V_{bond}, is symmetric about r_0, chemical bonds (in this case) can be treated as a simple harmonic function. The steep ascent produced by the model matches that of the potential energy profile for compressing a chemical bond, as described by the quantum mechanical model.

However, the spring model restricts a steep potential for bond stretching and does not allow the extension to long distances, where bond dissociation may take place. This means that the solutions obtained from this model are only approximations, and that the atoms within the simulated molecules are actually more tightly bonded than they really are. Therefore, the harmonic spring model is not applicable for the simulation of chemical reactions (such as those catalyzed by enzymes) where bond breakage and formation take place. However, in the cases of in which bond shifting is around the equilibrium, this model is a good approximation.

Bending Energy

In addition to being stretched, a bond can also be bent and twisted. The bent deformation can be thought as a deformation to the spring, similar to the bond length. Hence, the potential energy function V_θ for the bond angle can be treated in a similar manner to V_{bond} with θ_0 defining the equilibrium bond angle and k_θ the spring constant for the deforming of this angle.

$$V_\theta = k_\theta (\theta - \theta_0)^2 \qquad (2.4)$$

Torsion Energy

Imagine a string of four atoms (A, B, C, and D) linking to each other through a string of bond. **(**Figure 2.5**)**

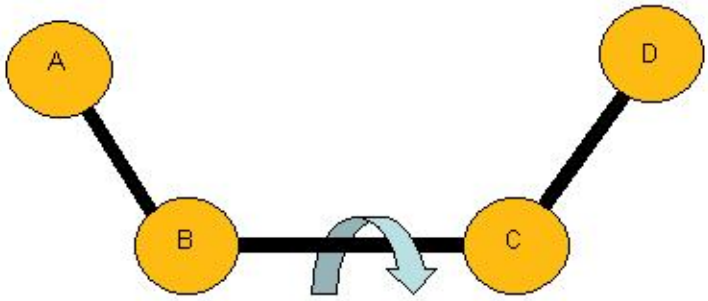


Figure 2.5 Typical torsion angle of bonded atoms

The twisting of a bond results in a dihedral angle ϕ about the central bond between atoms B and C of the four-atom center A-B-C-D. The potential energy function for ϕ (V_ϕ) depends on the type of the bond connecting B to C. Single bonds, for instance, are relatively free to rotate, while double bonds have very distinct energy minima at $\phi = 0°$ and 180°. In this case, V_ϕ can not treated as a simple harmonic springs function as with the previous cases, but, instead, takes the form of the periodic function: [17]

$$V_\phi = V_n \cos\left[\gamma + \frac{360°}{n}\right] \qquad (2.5)$$

V_n is the torsion force constant, n is the period of the function, and γ is the phase angle that defines the position of the minima.

Improper Torsions and Out-of-Plane Bending Motions

In addition to the standard terms mentioned previously, force fields can also include extra terms to describe molecular bonding in greater detail. An example of such terms is *out-of-plane* potential. One way to incorporate an out-of-plane potential into a force field is by using an *improper torsion* method. This method simply treats the torsion angle formed by four sequentially linked atoms as an 'improper' angle. [18] Its potential term can be expressed as a potential term, $V(\omega)$, at the 0° or at the 180° angle as followed:

$$V(\omega) = k\,(1\text{-}\cos(2\omega)) \qquad (2.6)$$

Another way of expressing out-of-plane potential is by calculating the angle and the height around the central atom. Suppose four atoms are arranged such that three of the atoms form a bond to a central atom. Two of the peripheral atoms and the central atom are within the same plane; while, the fourth atom is θ° above the plane. This θ° can be calculated and used as an out-of-plane potential. The calculation of the height can also be demonstrated using the same geometry. However, instead of calculating the angle, the distance, or the height (h), between the fourth atom from the plane is calculated. The angle and the height potential terms can be expressed as followed:

$$V(\theta) = \frac{k}{2}\theta^2 \qquad \text{and} \qquad V(h) = \frac{k}{2}h^2 \qquad (2.7)$$

Therefore, as we can see here, out-of-plane potentials can be used in a situation where a particular geometry (such as the planar geometry of an aromatic system) is desired or maintenance of the stereochemistry at a chiral center is necessary. [18] It is also important to realize that the methods mentioned above are only a few of the examples. One can design out-of-plane potentials whichever way he/she wishes. However, despite the usefulness of these terms, they are not always necessary. As a matter of fact, in some cases, the including of such terms may be deleterious to the performance of the force field. Vibrational frequencies, for instance, are particularly sensitive to the presence of out-of-plane terms.

Pyramid Height Terms

An alternative approach to using an improper torsion term to maintain chirality and planarity is to use a pyramid height term or triple product term. In this term the height of a central atom is restrained to be at a give distance above a plane defined by three other atoms. The cross product between two vectors defined by the three atoms in the base defines a unique

signed normal vector to the plane and the dot product of the vector between apical atom and an atom in the plane defines the height.

B. Nonbonding Potentials

The nonbonding potentials are those interactions that are not directly involved in covalent bonds, and they can be divided into two categories: the inter-molecular interactions and the intra-molecular interactions. The former refers to those interactions both between molecules and between the molecule and the solvent; while, the latter refers to the interactions between atoms or groups of atoms within a single molecule. The two types of interactions can also be break down further into electrostatic interaction, dipole-dipole interaction, van der Waals interactions, and hydrogen bonds.

The potential energy functions for the nonbonding interactions are inversely related to a power n of the distance, r, between atoms $1/r^n$. The range, at which a particular interaction becomes dominant, depends on n. For large r, $1/r^n$ approaches zero more rapidly for higher values of n. Conversely, for small r, $1/r^n$ approaches infinity more rapidly for higher values of n. Thus, functions that depend on high powers of r (where n is large) are called *short-range interactions*, while those with low powers of r (n is small) are *long-range interactions*.

Electrostatic Potential

The electrostatic potential V_e between two unit charges Z_1 and Z_2 can be calculated using Coulomb's law:

$$V_e = \frac{\left(Z_1 Z_2 e^2\right)}{Dr} \qquad (2.8)$$

The interaction is directly proportional to the product of the two charges (Z_1 and Z_2) and is inversely proportional to the dielectric constant of the medium D. r is the distance separating the two charges. V_e is used to define charges that are paired (*pair-wise*) interaction. Note that the

Coulomb's law does not explicitly account for shielding of the charges from counterions in solutions. However, this form of potential function is the form that is typically incorporated into the force fields.

An accurate treatment of the dielectric constant D in the electrostatic potential allows force fields to include the solvent effects on molecular structure without actually incorporating the solvent atoms into the model. [17] For example, two dielectric constants can be used to distinguish the interior and the exterior of a protein in solution. If the solvent is water, the value of 78.5 $\kappa\varepsilon^{\circ}$ is assigned for the dielectric constant at the exterior of the protein. The dielectric constant for the exposed atoms can also be set to a value similar to that of the solvent. However, the interior of the protein is usually treated as a low-dielectric cavity. The dielectric constant of the binding sites within metal-binding proteins, for example, usually has lower value than that of the solvent. This is due to the fact that lower dielectric constant is required for effective electrostatic interaction. Typical values for the dielectric constant for the interior of a protein range from 1 to 20 $\kappa\varepsilon^{\circ}$, with a good approximation being 3.5 $\kappa\varepsilon^{\circ}$. This is a rough estimate, since the true dielectric character must vary continuously throughout the molecule.

Dipole-Dipole Interactions

Two charges (a positive and a negative charges) separating at their centers with a distance $\boldsymbol{r}$ gives rise to an interaction called *dipole-dipole interaction*, which is characterized by a dipole moment. Using the distance between the two dipoles and the dielectric constant of the medium separating the dipoles can approximate the Coulombic interaction between two dipoles. However, the formulation of the dipole-dipole interaction depends upon the orientations of the two dipoles toward each other.

In many force fields, the dipole potential, V_{dd}, is incorporated into the potential function for electrostatic interactions by treating each atom as a monopole having a defined partial

valence. The Coulomb's law can then be directly used to evaluate the interaction. This approach has been successfully applied to charge-dipole interaction treatment, without the need for a separate function in the force field. The interaction between charges or permanent dipoles with induced dipoles is not normally treated in molecular mechanics force fields. Instead, induced dipole-dipole interactions are included in the van der Waals interactions.

Van der Waals Interactions

The best known form of the van der Waals potential functions is the Lennard-Jones 12-6 function. For two atoms, the Lennard-Jones 12-6 potential takes the following form:

$$V(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \qquad (2.9)$$

The Lennard-Jones 12-6 potential contains two adjustable parameters: the collision diameter σ (the separation for which the energy is zero) and the well-depth ε.

The Lennard-Jones potential is characterized by an attractive term, r^{-6}, and by a repulsive term, r^{-12}. The r^{-6} term is the same power-law relationship found for the leading term in theoretical treatments of the dispersion energy, such as the Drude model. [18, 19] However, there are no strong theoretical supports for the repulsive r^{-12} term. The twelfth power term is found to be acceptable for rare gases but is too steep for other systems, such as hydrocarbons. However, 6-12 potential is widely used, particularly for calculations on large system, as r^{-12} can be rapidly calculated by squaring the r^{-6} term. The r^{-6} term can also be calculated from the square of the distance without having to perform a computationally expensive square root calculation. [18]

Lennard-Jones equations with different powers for the repulsive potential have also been used. Values of 9 or 10, for examples, give a less steep curve and are used in some force fields. Lennard-Jones' original potential has been written in the following general form:

$$V(r) = k\,\varepsilon\left[\left(\frac{\sigma}{r}\right)^{n} - \left(\frac{\sigma}{r}\right)^{m}\right];\ \text{where } k = \left[\frac{n}{(n-m)}\right]\left[\left(\frac{n}{m}\right)^{\frac{m}{(n-m)}}\right] \qquad (2.10)$$

In an attempt of improving the potential without introducing more complexity from spectroscopy, Halgren has proposed an alternative functional form designed to be simple and easily incorporated into molecular mechanics calculations, while, improving the experimental reproductivity. [20, 21, 22] This potential has the general form:

$$V(r) = \varepsilon_{ij}\left[\frac{(1+\delta)}{(\rho_{ij}+\delta)}\right]^{(n-m)}\left[\frac{(1+\gamma)}{\rho_{ij}^{\,m}+\gamma} - 2\right] \qquad (2.11)$$

In this equation $\rho_{ij} = \dfrac{r_{ij}}{r_{ij}^{\,*}}$. The constants δ and γ apply to all interactions between the atoms ***i*** and ***j***. This potential reduces to the standard Lennard-Jones 12-6 potential if the following choice of parameters is used: $n = 12$, $m = 6$, $\delta = \gamma = 0$. Halgren proposed a *buffered 14-7* potential in which $n = 14$, $m = 7$, $\delta = 0.07$, and $\gamma = 0.12$, giving the following equation:

$$V(r) = \varepsilon_{ij}\left(\frac{1.07\,r_{ij}^{\,*}}{r_{ij}} + 0.07\,r_{ij}^{\,*}\right)^{7}\left(\frac{1.12\,r_{ij}^{\,*7}}{r_{ij}^{\,7}} + 0.12\,r_{ij}^{\,*7}\right) \qquad (2.12)$$

One of the reasons for developing the equation above was to keep the potential within a finite range, as the inter-atomic potential approaches zero. This is different from the previous Leannard-Jones function that allows the potential to become infinite, as the inter-atomic potential approaches zero. Another reason was that this form of the equation gives a more accurate reproduction of the series expansion for the dispersion interaction. Lastly, if a larger value of *d* is used then the repulsive component is greatly reduced without significantly changing the distance at which the potential crosses zero or the depth of the energy minimum. This feature is useful for optimizing structures with crude initial geometries. [18]

The r^{-12} term in the standard Lennard-Jones formulation has also been replaced by a theoretically more realistic exponential expression. Buckingham potential [18, 23], for instance, has the following potential form:

$$V(\mathrm{r}) = \varepsilon \left[\left(\frac{6}{\alpha - 6} \right) - \exp\left[\frac{-\alpha}{\left(\frac{r}{r_m - 1} \right)} \right] - \left(\frac{\alpha}{\alpha - 6} \right) \left(\frac{r_m}{r} \right)^6 \right] \qquad (2.13)$$

Note that a value of α between approximately 14 and 15 gives a potential that is close to the value given by the Lennard-Jones 12-6 potential in the minimum-energy region. However, when using the Buckingham potential, implementation must be carried out with care because the potential can become strongly attractive at very short distances. If this happens, a result of fused nuclei may occur during the calculation. Therefore, it is important for the program that uses this potential to able to keep track with the distance between atoms such that they are not becoming too close.

Another form of the potential was proposed by Hill [24]. This potential is sometimes referred to as the *Hill potential*. The Hill potential is an *exponential-6* potential with just two parameters: the minimum energy radius and the well depth:

$$V(\boldsymbol{r}) = -2.25\ \varepsilon \left(\frac{r_m}{r} \right)^6 + 8.28 \times 10^5\ \varepsilon \exp\left(- \frac{r}{0.0736 r_m} \right) \qquad (2.14)$$

The Hill potential was originally developed to enable the more realistic exponential term to be written in terms of Lennard-Jones parameters. The coefficients 2.25, 8.25×10^5, and 0.0736 were determined by fitting data for the rare gases and were assumed to be applicable to other non-polar gases. A Morse potential may also be used to make the van der Waals interactions in a force field, with appropriate parameters. [18]

Hydrogen Bonds

Another important interaction in bio-molecules is hydrogen bonding. Some force fields contain a separate potential term that describes the hydrogen bonds; while, others rely upon electrostatic and van der Waals interactions to reproduce hydrogen bonding.

One approach that has been used to describe hydrogen bonds in force fields is the replacement of the Lennard-Jones 6-12 term between hydrogen-bonding atoms with a 10-12 Lennard-Jones potential [18]:

$$V(\mathrm{r}) = \frac{A}{r^{12}} - \frac{C}{r^{10}} \qquad (2.15)$$

This function is used to model the interaction between the donor hydrogen atom and the heteroatom acceptor atom. Its use is intended to improve the accuracy with which the geometry of hydrogen-bonding systems is predicted. Other force fields incorporate a more complicated hydrogen-bonding function that takes into account the geometry deviations of both the donors and the acceptors, as a result of the hydrogen bond. The energy obtained from theses force fields are, therefore, coordinate-dependent.

YETI force field [25], for example, uses the following form for its hydrogen bonding term:

$$V(\mathrm{HB}) = \left(\frac{A}{r^{12}_{H-Acc}}\right) - \left(\frac{C}{r^{10}_{H-Acc}}\right)\cos^2\theta_{Don-H-Acc}\cos^4 w_{H-Acc-LP} \qquad (2.16)$$

The energy obtained from the above equation depends upon the distance from the hydrogen (H) to the acceptor (Acc), the angle subtended at the hydrogen by the bonds to the donor (Don) and the acceptor, and the deviation of the hydrogen bond from the closest lone-pair (LP) direction at the acceptor atom.

Another program called the GRID program [26] finds energetically favorable regions in protein binding sites using a direction-dependent 6-4 function:

$$V(\mathrm{HB}) = \left(\frac{C}{d^6} - \frac{D}{d^4}\right)\cos^m \theta \qquad (2.17)$$

θ is the angle subtended at the hydrogen and m is usually set to 4.

Cross Terms

Incorporating cross terms into a force field significantly improves structural simulation. One reason is because their contribution enables the force field to reproduce experimental data, such as the vibrational spectra, of molecules more accurately. Hence, these terms are often defined in force fields to optimize their performances.

Most cross terms are functions of combined terms, such as stretch-stretch, stretch-bend and stretch-torsion. More complicated terms, such as the bend-bend-torsion, can also be included. The functional forms for cross terms can be formulated in various ways, depending upon the purpose. For example, the stretch-stretch cross term [18] between two bonds 1 and 2 can be modeled as:

$$V(l_1,l_2) = \frac{k_{l_1,l_2}}{2}\left[(l_1 - l_{1,0})(l_2 - l_{2,0})\right] \qquad (2.18)$$

Similarly, the stretching of the two bonds adjoining an angle can be modeled using an equation of the following form (as in MM3, MM3, and MM4):

$$V(l_1,l_2,\theta) = \frac{k_{l_1,l_2,\theta}}{2}\left[(l_1 - l_{1,0}) + (l_2 - l_{2,0})\right](\theta - \theta_0) \qquad (2.19)$$

A stretch-torsion cross term can be used to model the stretching of a bond that occurs in an eclipsed conformation. Two possible functional forms are:

$$V(l,w) = k(l\text{-}l_0)\cos(nw) \qquad (2.20)$$

$$V(l,w) = k(l-l_0)\,[1+\cos(nw)] \qquad (2.21)$$

where n is the periodicity of the rotation about the bond ($n = 3$ for sp^3-sp^3 bonds).[18] In addition to the cross terms mentioned above, terms describing torsion-bend and torsion-bend-bend terms may also be formulated and included in force fields.

Although, cross terms play significant roles in simulation of molecules, adding cross terms for all contributions to the force field is not necessary. In fact, it is found that only a few cross terms are needed in order to reproduce structural properties accurately. Maple, Dinur and Hagler [27] used quantum mechanics calculations to investigate which of the cross terms are most important. They suggested that that stretch-stretch, stretch-bend, bend-bend, stretch-torsion, and bend-bend-torsion were most important [28].

It has also been suggested that the cross terms (together with some other features) can provide a general way to classify force fields [29] into three classes. Class I force fields are the simplest because they only contain harmonic terms and no cross terms. Class II force fields, on the other hand, are more advanced. In addition to those components found in Class I, the Class II force fields also include anharmonic terms and cross terms to account for the coupling effects between coordinates.[18] These Class II force fields are, therefore, capable of predicting the properties of unusual systems, as well as reproducing the vibrational spectra of the simulated molecules. The Class II force fields can also be used (without modification) to model the properties of isolated small molecules, of systems within condensed phases, and of macromolecular systems. [18]

Class III force fields, such as the MM series force fields developed by Allinger [30], are the most advanced force fields that are currently available. They take into consideration of chemical effects, electronegativities, and hyperconjugation. One example of the hyperconjugation effect is the change in the length of the C-H bond in acetaldehyde with rotation

about the C-C bond. [18] When the C-H is perpendicular to the plane of the carbonyl group, there is maximum overlap between the σ orbital of the C-H bond and the π* orbital of the carbonyl carbon. As a result of the migration of the electron density from the C-H bond to the π* orbital, the C-H bond becomes longer, and there is also a greater contribution from the charged resonance structure. On the other hand, when the bond to the hydrogen atom is within the plane, the overlapping is minimized. *Ab initio* calculations have suggested that the bond length changed by 0.006 Å between the two forms. This effect was incorporated within MM4 by a term of following form:

$$\Delta l = k(1 - \cos 2w) \qquad (2.22)$$

Although there is no hard evidence on the existence and origin of the hyperconjugative effects, low-temperature X-ray crystallographic experiments on appropriate compounds, together with *ab initio* calculation, indicate the presence of hyperconjugative effect in a detectable level. [18]

2.2.2 Atom Types and Parameterization

Structural simulation requires more than just accurate potential equation. When preparing for structural simulation, it is necessary to supply the simulating program with some information about the target molecule. This information may include the atomic numbers of the nuclei present, the geometry of the system, and the overall charge and spin multiplicity. Other important information that is often included in molecular force field is atom's hybridization state and, in some cases, the local environment. However, one needs to keep in mind that these properties are not fixed and that they can vary as the geometry and the environment of the atom change.

Designing a computer program to deal with the molecular flexibility is a challenge. Many factors must be taken into consideration before the final conclusion can be drawn. To bypass this complication, a specific *atom type* is assigned to the atoms, and the associated atomic properties are attached to the atom in a *parameterized* fashion. Each force field has its own set of atom types and parameters. For instance, the MM2, MM3, and MM4 force fields (developed by Allinger and co-workers) [30, 1, 31, 32] are widely used for calculation on small molecules. These force fields can distinguish the following types of carbon atom: sp^3, sp^2, sp, carbonyl, cyclo-propane, radical, cyclo-propene, and carbonium ion. In the AMBER force field (developed by Kollman and co-workers) [33, 34], the atoms are further differentiated to account for the differences in their chemistry. For example, AMBER assigns a different atom type (that is different from the carbon atom in an isolated five-membered ring such as histidine) to the carbon atom at the junction between a six- and a five-membered ring. Hence, AMBER is able to distinguish this fused carbon from the regular carbon atoms in a benzene ring. The AMBER force field can also assign different atom types to a histidine amino acid, depending upon its protonation state. [18] In contrast to the specific force fields such as MM4 and AMBER, more general force fields generally assign these carbons to the standard 'sp2 carbon' atom type. Hence, the choices of which force field to use depend on the type of molecule being simulated.

Although there seems to be force fields available for all types of molecule, each of these currently available force fields is only applicable to specific type of molecules. One current research in this field is the development of a force field that can be universally applied to the simulation of all bio-molecules. The AMMP SP4 force field (a force field developed by Harrison [35, 36], for example, is under development for such a purpose. The goal of the project is to improve the simulation potentials so that all bio-molecules (including both macro- and micro-

molecules, as well as those associated with metals) can be accurately simulated in under all conditions. The relative performance of different force fields is shown in Figure 2.6.

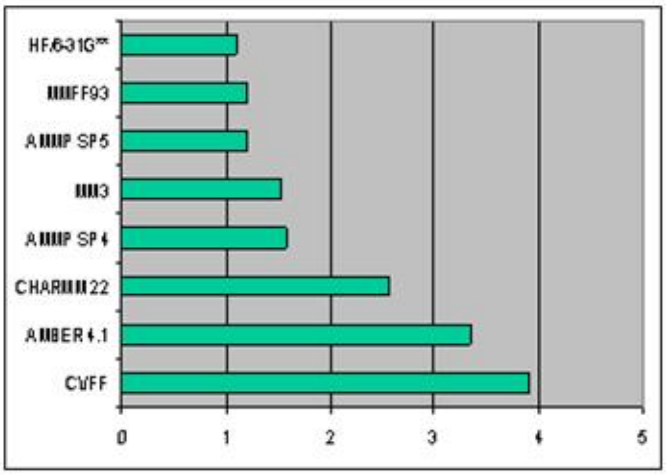


Figure 2.6 Relative Performances of Different Force Fields. The relative performance of different potential sets is shown in this figure which shows the RMS error (in kcal/mol) for several sets on a peptide benchmark [37, 38]

2.2.3 Currently Available Force Fields

There are a number of force fields currently available. All of them have undergone through a number of tests to ensure their accuracy in molecular simulation. However, due to the scope limitation, this review will only give an overview of four force fields: the MM force fields, AMBER, CHARMM, and UFF. For more detail descriptions of the force fields, please refer to Schlecht, 1998 [39].

A. The Allinger MM Force Fields

Allinger, N.L. and his research group were the developers of the MM series (MM1, MM2, MM3, and MM4) force fields. MM1 was one of the first force field developed for the simulation of hydrocarbon compounds, and its potentials were only designed to work with two atom types: hydrogen and sp^3-hybridized carbon. [40] This lead to the development of the improved version of the force field, the MM2 force field. This new force field was first published in 1977 and

quickly become of the widely used force field in organic chemistry. MM2 added many more atom types, as well as added the electrostatics and the out-of-plane angle bending terms into its force field. [41] Because of these new features, MM2 was intended for the use in calculations of conjugated system.

It was over two decades before MM3 was released. One of the motivations that lead to the development of MM3 was the desire to produce more accurate predictions of molecular vibrational spectra. The MM3 force field was designed to incorporate many of the advanced features of the MM2, while, at the same time, compensate for the deficiencies existed in MM2. These deficiencies included: the underestimation of high C-C rotational barriers in congested hydrocarbons, overestimation of hydrogen-hydrogen nonbonded repulsions at very short distances, underestimation of the bond length when the bonds are in the eclipsed conformation, the failure to model the stability of the perpendicular benzene dimmer vs. the face-to-face dimmer, and difficulties with the angle bending potential at large deformations, specifically with three-, four-, and five-membered rings. [42]

MM3 contains more detailed potential expressions for bond stretching and angle bending. The force field also contained a modified van der Waals potential, as well as several additional cross terms. The parameterization for three-, four-, and five-membered rings has also been expanded, and five-membered-ring carbon atom types were introduced. [43] However, the use of explicit lone pair pseudo-atoms was discontinued. Electrostatics for uncharged species is still calculated on the dipole-dipole interaction model. Many individual applications of the MM3 force field have been reported, such as aldahydes and ketones, [42] alkyl iodides, [34] etc. MM3 has 152 parameterized atom types, including for example 9 types of hydrogen, 21 types of carbon, 21 types of nitrogen, 46 types of oxygen (many of these in various acid derivatives, 8

types of sulfur, 5 noble gases, a large number of metals, and a dummy atom, and the lone pair of electrons has been removed from the atom list. [43]

The most recent version of the MM force field series is MM4, which was released in 1996. [44] MM4 employs the same library of atom types as found in MM3, with a few additions. MM4 was designed to address several specific deficiencies in MM3. These include inaccurate calculation of the vibrational spectra, low rotational barriers for sterically congested molecules, and the exclusion of the vibrational energy in the heat-of-formation calculations, etc. [45, 46]

B. AMBER Force Field

The AMBER (Assisted Model Building with Energy Refinement) molecular modeling program and its force field were developed by the Kollman research group. [47] The program has continued to evolve primarily under the guidance of the Kollman group, but with contributions from others as well. [48, 49] AMBER was one of the first force field designed primarily to model bio-macromolecules such as proteins and nucleic acid polymers. However, the force field can also handle a wide variety of other molecules. Although AMBER models bonds and angles with simple harmonic potential without cross terms, its treatment of electrostatic interactions is more sophisticated than some of the force fields designed primarily for small and relatively non-polar molecules.

AMBER contains three types of internal libraries: proteins, nucleotides, and other. During the initiation of the simulation process, the target molecule is first identified by the program as 'P' for proteins, 'N' for nucleotides, or 'O' for other types of atoms. If the molecule falls into either one of the first two cases, the program will access the internal library of amino acid or nucleotide fragments. [39] AMBER uses tree data structure to represents a molecule. Descriptors are used to define functional groups. Hence, based on the assigned descriptor values, the program can also recognize those structures, which do not fit the definition of the tree.

AMBER also contains a modified conjugate gradient minimizer for the energy optimization, and gives the user a choice between gas-phase and solution-phase potentials. Usually when the solution potential is used, the charged interactions are reduced, and the van der Waals interactions of hydrophobic atoms are modified.

There are two versions of AMBER force fields. The first version, or the *united atom version*, had 40 parameterized atom types. These include: eighteen carbon types, eight nitrogen types, four oxygen types, six hydrogen types, two sulfurs, a phosphorus, and a lone pair pseudo-atom. The second version, or the *all atom version*, added five more sp^2 carbon types, in order to address non-bonded interactions between aromatic groups. There are no *sp*-hybridized atom types.

C. CHARMM Force Field

CHARMM is an acronym for "Chemistry at Harvard Macromolecular Mechanics". It is a molecular modeling program developed by M. Karplus and coworkers. This advanced program is capable of carrying out static energy and dynamics calculations on protein, nucleic acids, prosthetic groups and substrates, polypeptides, and other related macromolecules. [50-56] The program can also be used to study static energy, molecular dynamics, salvation, electrostatics, crystal packing, vibrational analysis, free energy perturbation, combined quantum mechanics and molecular mechanics, stochastic dynamics, etc.

CHARMM has three methods to model hydrogen atoms: (1) the *all atom method* that treats all hydrogen atoms as discrete atoms, (2) the *extended atom method* (equivalency of the united model for AMBER) that uses atom types for a combined heavy atom and its hydrogens, and (3) the *intermediate atom method* which uses both the extended and the all atom methods to represent the hydrocarbons that are not parts of functional groups and their hydrogens, respectively.

D. UFF Force Field

Universal Force Field, or UFF, was first developed by Rappe *et al.* [57-60] UFF simulates molecules based on the uses of general force constants and geometry parameters derived from atoms' hybridization and connectivity. The information required for deriving these parameters are usually presented in the input structural file, or can be extrapolated from the data of closely related systems. UFF atom types cover the entire periodic table, as well as the different oxidation states of the transition elements. The atom types are denoted by a five-character code: the first two spaces are for the atomic symbol, the third space is a number or a letter representing the hybridization or the geometry of the atom, and the fourth and fifth spaces are reserved for other parameters such as formal oxidation state. UFF determines the natural bond length between atoms by summing up the contributions from each of the two atoms, the bond order correction, and the electro-negativity effect correction. A single value is used for the bond dissociation energy component of the Morse potential, multiplied for multiple bonds. The bond stretch force constants are derived from the generalized Badger's rules, [61-64] as a function of the effective charges on each atom.

2.2.4 Force Fields Performances

Despite the wealth of information on molecular dynamic simulation, there are only a few literatures that compare the performance of the force fields. One of the recently published works by Mu *et al.* compares the simulation performance of various force fields in the simulation of tri-alanine in water. [64]

A number of experimental studies have been performed on tri-alanine. [65-71] Spectral studies [68-70] suggested that trialanine mostly adopts a conformation around (ϕ, ψ) ~ (-60°, +140°), which is known as poly(Gly)II or P_{II} structure. [66, 67] The dihedral angles of the

peptide has also been observed (to a smaller amount than P_{II} structure) around ~ (-80, -50) and ~ (-120, +130), which represent the regions of right–handed helix α_R and the extended β conformations [71], respectively.

In his study, Mu selected six popular force fields: parm94.dat (A94) and parm96 (A96) of AMBER, CHARMM version 22 (C98), a standard set 43A1 (G96) and a refined set 45A3 (G01) from GROMOS, and OPLS (O96). For all of the selected force fields, 20 ns long MD simulations is performed to study in detail the structure and conformational dynamics of the solvated peptide. Based on the computational studies, Mu concluded that trialanine populates predominantly (>80%) the P_{II} state, to a smaller amount (< 20%) the α_R state, and only insignificantly (<5%) the extended β state.

The results obtained from A96, G96, G01, and O96 predict that solvated tri-alanine is formed in an equal parts of the extended conformations β (~40%) and P_{II} (~40%) but only (~20%) in the helix conformation α_R. Although the overall ratio of extended and helical population is in agreement with experimental evidence, these force fields clearly oversample the β conformation. On the other hand, A94 and C98 force fields predict a relatively large population of the helical α_R conformation, whereas the β conformer is hardly populated. Here the latter result is in accordance to experiment, but the α_R is clearly oversampled relative to the P_{II} state.

Mu compared the solvated results to those of the isolated trialanine, using the same set of force fields. Surprisingly, the results from A96, G96, G01, and O96 also favor the β state over the α_R, as found in the solution. Similarly, A94 and C98 show a favor toward the α_R conformation over the β state in the gas phase, which is also consistent with the results obtained

from the solution peptide. Based on this observation, it appears that the performance of the force fields in solution depends strongly on their properties in the gas phase.

Based on his observation, Mu suggested that the way these new force fields were derived or modified from their parent force fields may play a part in this population ratio difference. In the case of the two force fields of AMBER, the A96 model was derived from the A94 model through a re-parameterization of the backbone dihedral parameters, which were adjusted to reproduce the gas-phase energy difference between extended β and constrained α_R helical conformations for the alanine tetra-peptide, as calculated by high-level *ab initio* methods. The two GROMOS force fields G96 and G01, on the other hand, differ in their parameterization of the aliphatic carbons, which was found to have only minor effects on the population of the conformational states of trialanine.

According to Mu, the accuracy of the simulation outcome typically depends on free-energy differences of less than $k_BT = 2.5$ kJ/mol, which is hard to achieve by an empirical force field. As a remedy, Mu suggested the use of microscopic MD description with the correct statistical weights obtained from experiments. For the AMBER and GROMOS force fields, the weights were designed such that > 80% was assigned to the P_{II} state, approximately 20% was assigned to the α_R state, and the β conformation was discarded. When this is done, virtually all available experimental data can be reproduced with high accuracy.

Another force field comparison is done by Swarnalatha *et al.* to evaluate the performance of the following revised force fields: AMBER4.1, BMS, CHARMM22, and CHARMM27 for the simulation of the minor groove of DNA. [71] CHARMM27 is the latest version of the CHARMM force fields. It is designed to correct the over-stabilization effect of the A form DNA over the B form DNA that exists in CHARMM22. CHARMM27 contains small significant changes in both the internal and interaction parameters relative to CHARMM22. Based on the

obtained data, this new CHARMM force field seems to work well in equilibrating between the A and the B forms of DNA under the influence of the environment, such as the water activity. Similarly, the revised AMBER force field, AMBER4.1, has also been shown to produce results that are better agreed with experimental data, as a result of the internal parameter adjustments. Both CHARMM27 and AMBER4.1 force field parameters are derived based on their ability to reproduce experimental results of nucleic acid oligomers, as well as on their consistency with results obtained from quantum mechanical calculations of small molecules. Bristol-Myers Squibb (BMS) force field, by Langley, was developed, in part, by adaptation of the CHARMM22, QUANTA, and AMBER force fields.

According to the simulation performed by Swarnalatha, the results show that the average simulated structures of the d(CGATTAATCG)$_2$ DNA double helix obtained with the CHARMM27, AMBER, and BMS force fields have a B-form geometry close to the x-ray structure; while, for CHARMM22, the starting B-DNA structure undergoes a transition to an A-like DNA structure. This result shows that all three of the new force fields are clearly more efficient for simulation studies of DNA than CHARMM22. However, out of the three force fields, BMS seems to yield results that are most agreed with the x-ray data. The remaining two force fields yield similar results, each of them better describes certain structural features of the B-DNA.

As for conformational flexibility, the results obtained from AMBER4.1 and CHARMM27 are more consistent with the available experimental data than those obtained from BMS. The salvation of the minor groove simulated by AMBER4.1 and CHARMM27 contains ions in the primary shell, which is consistent with the experimental data and previous DNA simulation; while, there are no ions observed in the primary salvation shell of the minor groove simulated by BMS. According to Swarnalatha, the lower calculated conformational flexibility of the B-DNA

with BMS might be responsible for this, because expanding of the minor groove is necessary to accommodate some ions. Alternatively, the energetic penalty associated with the partial dehydration of the counterions might be too high to allow the penetration of counterions into the minor groove. However, the BMS force field gives the strongest salvation of nucleic acid bases among all the force fields.

The size of the minor groove of the simulated DNA is also compared in this experiment. The obtained results show that the size of the minor groove simulated by the three force fields tends to be nonspecifically wider than that of the x-ray structure. However, some local narrowing of the minor groove is observed. According to Swarnalatha, the narrowing of the minor groove is generally associated with the presence of couterions in the minor groove. The presence of the divalent cation (Mg^{2+}) at the TpT basepair step leads to a local widening of the minor groove in the x-ray structure. In the DNA simulation with AMBER4.1 and CHARMM27, the substitution of the divalent cation by the monovalent cation induces instead of local narrowing of the minor groove. In the absence of any counterion at the position of the original magnesium binding site, the minor groove does not show any narrowing with the BMS force field.

2.2.5 Choosing Force Fields

There are a number of force fields currently available for molecular simulation, and most of them are proven to yield structures and molecular properties that are consistent with the experimental data. However, there is no "best" force field currently available for all problems. Some force fields are specifically designed to deal with particular types of molecules, such as nucleotides and proteins; while, others are designed to be more general problems, such as simulation of organic compounds in general. The force fields that fall into the first category are called *instance-centric* force fields, and the ones that fall into the second category is called *atomic-centric* force fields.

CHARMM, AMBER, and OPLS are examples of instance-centric force fields. They are designed to include properties of specific systems, particularly those of nucleotides and amino acids. The simulation of the macromolecules can be performed, with compatible results to the experimental data. They can also be improved and/or manipulated, if desired, easily as long as the target system is a part of the original system. This inflexibility is a weakness of these force fields. Using these force fields for the simulation of other types of molecules must be evaluated carefully and monitored closely with those of experimental data.

On the other hand, force fields like MM3, UFF, and SP4 are more flexible, which is an advantage of atomic-centric force fields. Their design is based on the geometry of atoms in the molecule, instead of on the types of molecules. Therefore, these force fields can be applied to a wider range of molecules than those of instance-centric force fields. However, the weakness of atomic-centric force fields is that the obtained results may not be as accurate as those from obtained from the instance-centric force fields. Because of their generality, specific properties such as those of macromolecules may be masked. Hence, the choice of which force field to use depends on the particular system of interest. If the target system is specific, then instance-centric force fields may be preferred over the atomic-centric force fields, and vice versa. Another consideration that should also be taken, particularly with newer derived force fields, is the reliability. Some force fields have been tested with a wide range of molecules, and they can be more reliable than those which have only been used in a few specific studies.

2.3 Energy Minimization

A typical process of minimization is illustrated for a diatomic molecule in Figure 2.7.

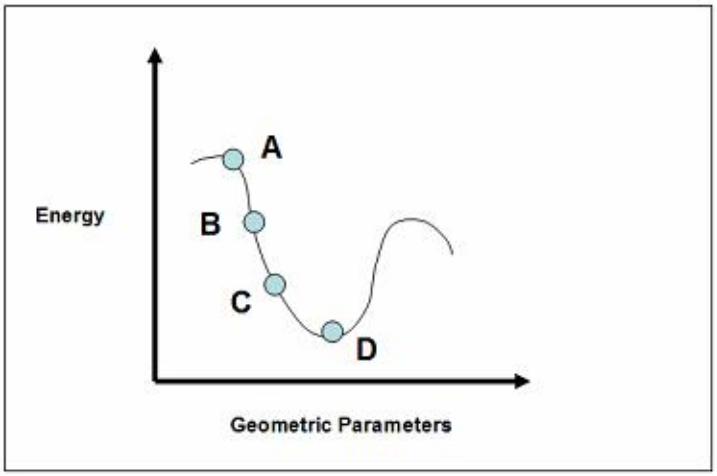


Figure 2.7 Typical Energy Minimization Scheme

The wavy line in Figure 1.10 is the surface energy of a molecule. Each point on the surface represents a conformer that has a specific amount of energy (y-axis). Basically, the initial structure at the position A can undergo modification such that a new conformation (either to the left or to the right of position A) with lower energy can be achieved. In this case, moving to the right lowers the energy, and the initial position is forgotten. The process repeats at B, then at C, and finally at D. The process continues until all possible small changes to the structure increase the energy of the structure. In this case, once at position D, all possible small changes will increase the energy of the system. Hence, conformation at point D in this example is the most stable conformer. This process is straight forward for a one-dimensional problem. However, bio-molecules in the biological system have $3N$-6 degree of freedom, where N is the number of atoms in the molecule. This means that not only the searching for the lowest-energy structure becomes more complicated, but also the total time of all energy calculations for the system significantly increases.

To solve this problem, a modification is made to the traditional minimization strategy. Instead of picking a starting point, a set of points (A and C in Figure 2.8) are selected as initial

structures. The conformations of the initial structures are allowed to change, in a 'roll downhill' fashion, until their nearest lowest-energy conformation (NLEC) is found.

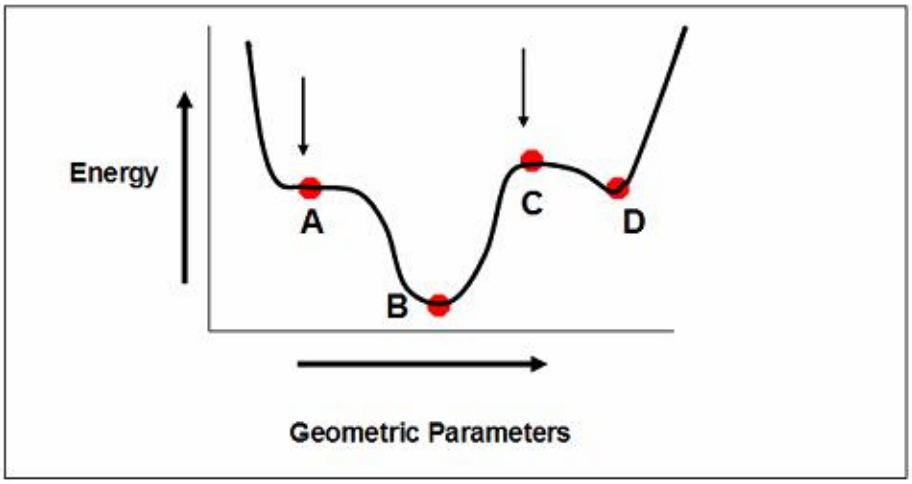


Figure 2.8 The modified scheme for finding energy minima on the potential energy surface

The energy levels corresponding to the conformers at points A, B (for now), and D in Figure 2.8 are known as *local energy minima*. These energy minima are then compared, and the lowest of all (conformer at point B) is known as the *global energy minimum* of the structure. The conformation corresponds to the global energy minimum energy is assumed to be the most stable conformation. It is worth noting that, although most minimization algorithms can only go downhill on the energy surface, some specialized minimization methods can make uphill moves to seek out minima lower in energy than the NLEC. However, no algorithm has yet proved capable of locating the global energy minimum from an arbitrary starting position.

Minimization methods or algorithms can be categorized into two groups: those which use derivatives of the energy with respect to the coordinates (derivative methods) and those which do not (non-derivative methods). Non-derivative methods do not require the calculation of the derivative (the slope) of the potential energy surface. As a result, their programs are easier to design, but are generally less efficient. In this review, we will, therefore, focus only on the

derivative methods because they provide more detail information on energy minimization and are commonly used by the minimization methods.

2.3.1 Derivatives

Derivatives are important for two reasons: (1) they provide information about the shape of the energy surface, and (2) they can significantly enhance the efficiency of locating the minimum on the energy surface. In energy minimization, the first and the second derivatives are the most important. The direction of the first derivative of the energy, or the *gradient*, indicates the location of the minimum, and the magnitude of the gradient indicates the steepness of the local slope. This allows atoms to move in a response to the applied force such that the new conformation is lower in energy than the previous conformation. Second derivatives indicate the curvature of the function. This information can be used to predict where the function will change direction, such as pass through a minimum or some other stationary point.

The function of energy minimization can be written as a Taylor series expansion about the point x_k.

$$V(x) = V(x_k) + (x - x_k)\, V'(x_k) + (x - x_k)^2 \frac{V''(xk)}{2} + \ldots \qquad (2.23)$$

For a multidimensional function, the variable x is replaced by the vector $\boldsymbol{\chi}$ and matrices are used for the various derivatives. Thus if the potential energy $V(x)$ is a function of $3N$ Cartesian coordinates, the vector $\boldsymbol{\chi}$ will have $3N$ components and $\boldsymbol{\chi}_\mathbf{k}$ corresponds to the current configuration of the system. $V'(\boldsymbol{\chi}_\mathbf{k})$ is a $3N \times 1$ vector matrix that contains elements of which are the partial derivatives of V with respect to the appropriate coordinate, dV/dx_i. Each element (i, j) of the matrix $V''(\boldsymbol{\chi}_\mathbf{k})$ is the partial second derivative of the energy function with respect to the two coordinates x_i and x_j, $\dfrac{d^2V}{dx_i dx_j}$. $V''(\boldsymbol{\chi}_\mathbf{k})$ is thus of dimension $3N \times 3N$ and is known as the

Hessian matrix or the *force constant* matrix. The Taylor series expansion can be written in the following form for the multidimensional case:

$$V(\chi) = V(\chi_k) + (\chi - \chi_k)\, V'(\chi_k) + (\chi - \chi_k)^T \tfrac{1}{2} V''(\chi_k) + \ldots \qquad (2.24)$$

However, the energy functions used in molecular modeling rarely behave as pure quadratic and, therefore, the Taylor series expansion can only be considered an approximation. This implies that, first of all, the performance of a given minimization method will not be as good for a molecular mechanics or quantum mechanics energy surface as it is for a pure quadratic function. Second, the harmonic approximation is poor at those locations that are far from the minimum. Hence, less robust minimization methods usually fail at these positions, even though they may work very well close to a minimum, where a harmonic approximation is more valid. However, a drawback of robust minimization protocols is that they are usually less efficient than the less robust protocols. One possible approach to handle this problem is to design an algorithm that combines the two protocols such that the strength of the minimization can be adjusted according the location on the energy surface.

The derivative methods can be classified according to the highest-order derivative used. First-order methods use the first derivatives (i.e. the gradients) whereas second-order methods use both first and second derivatives. The simplex method can be considered a zeroth-order method as it does not use any derivatives.

A. First-Order Minimization Methods

Two first-order minimization algorithms that are frequently used in molecular modeling are the *steepest descent* and the *conjugate gradient* methods. In these methods, the coordinates of the atoms change gradually as the system moves closer to the minimum point. The starting point of each iteration (k) is the molecular configuration obtained from on the previous step, which is

represented by the multidimensional vector x_{k-1}. For the first iteration the starting point is the initial configuration of the system provided by the user, the vector x_1.

Steepest Descent Algorithm

This mathematical method was first applied to molecular mechanics for discovering a minimum-energy molecular structure by K. Wiberg. [72] In this method, the initial energy is calculated from the input geometry. Following the initial step, each coordinate is changed by a small amount, and the resulted energy is evaluated. The derivative of the energy change is then calculated, and that coordinate is returned to its original value. This process repeats with the next coordinate and continues until all of the coordinates have been examined.

From this data, the vector direction of steepest descent of the energy for each atom is determined. The atoms are then moved together along their respective vectors of steepest descent, and this process is reiterated as long as the energy continues to decrease. If further travel along a particular vector begins to increase the energy, the movement of that tom along that vector is reversed.

When the change in energy between iterations drops below a threshold level, the resulting structure is assumed to be at an energy minimum. [39] The steepest descent method converges rapidly when far from the equilibrium position, but the approach slows as the atoms draw closer to their equilibrium positions.

Conjugate Gradient Algorithm

This mathematical method was first applied to molecular mechanics stain energy minimization by Engler, Andose, and Schleyer. [73]

As in the steepest descent, the conjugate gradient algorithm performs a line search in a direction largely specified by the gradient of the energy. Unlike steepest descent the search direction includes earlier search directions in order to ensure that the search covers a larger region

of space. The movement along one vector does not accidentally undo the progress towards the minimum established along another vector.[39] Consequently, the conjugate gradient method converges more rapidly than does the simple steepest descent method. The search direction is specified by the current gradient plus a scaled copy of the previous search direction.

Not all implementations of the conjugate gradients algorithm are equivalent in performance. Several numerical 'tricks' will improve the performance of an implementation. In general, it is probably best to use a variant of the Polak and Ribiere β because this formulation does not require an exact line search [74]. Therefore the conjugate directions property holds even when approximate solutions are found along search directions. Experimentally we have found it advantageous to use an inexact line search. When the target function is highly curved and possesses many local false minima due to floating point errors, using an inexact line search keeps the search from converging too soon. It is also useful to monitor both the average square (l^2 norm) of the force as well as the maximum force on any atom (l^∞ norm) as a true minimum will have a minimum in both norms.

B. Second Derivative Methods

Newton-Raphson Algorithm

This method was first applied to molecular mechanics strain energy minimization by Boyd using the method of numerical differentiation. [75] This method applies both the first and second derivative of the change in energy in order to guide the system to equilibrium.

For N atoms, the movements of each of which have components along the X-, Y-, and Z-axes, the Newton-Raphson method involves calculating a matrix of second derivatives of dimensions $3N \times 3N$. This is referred to as the *Hessian matrix*. In a geometry optimization, only $3N$-6 degrees of freedom are considered, since three translations and three rotations will not

affect the energy of the system. After these nonproductive degrees of freedom are subtracted, the matrix is inverted (*inverse Hessian matrix*) to obtain lower-energy geometry. This process is repeated until convergence to a local minimum.

The Newton-Raphson algorithm performs best on harmonic energy surfaces near the equilibrium point, but does poorly when the system is distant from its equilibrium point. The second derivatives are also computationally more intensive.

Block-Diagonal Newton-Raphson Algorithm

This method was first applied to molecular mechanics strain energy minimization by Allinger and coworkers. It is a simplification of the Newton-Raphson algorithm, which avoids the computational load of calculating the full matrix by operating only upon the 3×3 portion of the matrix for each atom. These sub-matrices are 3×3 blocks which lie along the diagonal portion of the full matrix, which gives rise to the "block-diagonal" portion of the name. For a molecule of N atoms, this method requires only $9N$ elements. In comparison with the full matrix Newton-Raphson method, the computer time per iteration will be much less with the block-diagonal. However, the minimization process may require more iterations with the block-diagonal method, because the geometry makes less progress toward equilibrium during each iteration. This method is also known as *non-simultaneous local energy minimization*. [76]

Quasi-Newton Methods

The two Newton's methods are very attractive, except for the need to calculate second derivatives. One major drawback in obtaining second derivatives using these methods is the time-consuming operation in the calculation of the inverse Hessian matrix. This could be a serious drawback in practice, unless the second derivatives are easy to calculate. The Quasi-

Newton methods (also known as *variable metric methods*) alleviate this problem by gradually building up the inverse Hessian matrix in successive iterations. [77]

The following is the formula shows the relationship between positions and the inverse Hessian matrix:

$$X_{k+1} = x_k - H_k g_k \qquad (2.25)$$

where H_k and g_k are the Hessian and gradient at iteration k. Basically, the above equation states that at each iteration k, the new position X_{k+1} are obtained from the current position x_k, the gradient g_k and the current approximation to the inverse Hessian matrix H_k.

For the specific case of iteration in the above equation, the key question is how to include curvature information, as the one carried by Hessian, while not computing the Hessian at every iteration. A class of methods called Quasi-Newton methods build up curvature information by observing the behavior of f_k and g_k during a sequence of descent iterations. A sequence of points is used to generate an approximation to the Hessian, rather than evaluating the Hessian at a single point. Quasi-Newton methods today are regarded as by some people as the best methods for solving unconstrained problems, however the difference in performance between Quasi-Newton methods and conjugate gradients is not large and large problems can only be handled by conjugate gradients because conjugate gradients does not require the storage of an approximate Hessian.

2.3.2 Heuristic Methods for Global Minima Searching

A. Simulated Annealing

Simulated annealing method, which was first described in 1983 [78, 79], is a method designed to search for the global minimum of a potential surface. Simulated annealing algorithms often work by making random changes to a structure. The newly formed structure is

then determined if it is better than the previous structure, using Metropolis criterion. Simulated annealing starts with a random configuration and random velocity for each atom. The program then traces the movement of the molecule as the energy is slowly removed. This procedure is very much like real annealing, and it is guaranteed to work, provided the energy is removed infinitely slowly. A more rapid rate of cooling will remove the certainty of success, but this approach to finding global minima seems to work quite well. This sort of approach is often used to optimize structures generated by distance geometry techniques.

B. Genetic Algorithm

Genetic algorithms, based on the ideas drawn from the fields of genetics [80], are mathematical techniques designed to deal with problems in optimization. The system which needs to be optimized is described in terms of a "chromosome", which is a string of numbers. For conformation searching problems, this could be a list of torsion angles. For a biological system, this would correspond to a base pair. An initial population of conformations is generated either by using random numbers or by careful design. This will be the first generation. The next generation is created from the first-generation conformers by mixing and mutating the information in the chromosomes. However, it must be aware that too much alteration will destroy the useful information in the pervious generation; while, too little will make it impossible to explore new areas of conformation space. This mixing is achieved by a method of crossover, which takes two chromosomes, chops them into two, and recombines the pieces to make two new chromosomes. The two new chromosomes must both be the same length as the parents. The mutation is simply a random change in a few of the numbers in a few of the chromosomes. Once these two methods have produced new chromosomes, the best new group of chromosomes is chosen to be the next generation.

The criteria for choosing the best may not be easy to define. In a conformation search, one of the chromosomes will have the lowest energy, and so this may be regarded as the best chromosomes. The best generation will certainly not simply be several copies of this chromosome, however, because diversity is needed to explore new areas of conformation space. The performance of a genetic algorithm will depend on the choices for the size of each generation, the rate of crossover, the rate of mutation and the method of selecting a new generation. Each of these choices can be made in different ways, and the optimal strategy is not clear. However, genetic algorithms have been shown to be useful in conformation search [81-83], particularly in the study of structures for which each entry in the chromosomes has a similar significance, such as unbranched alkanes [84].

C. Molecular Dynamics

A molecular dynamics simulation begins with giving each atom in a molecule some kinetic energy. This allows molecular motions of the molecule to be calculated by solving the Newtonian equations of motion. The information obtained from solving the motion equations can be used to analyze the current status of the molecule and to predict what will "probably" happen in a very short time in the future. The calculation, however, is a difficult one because it requires a lot of computer power to simulate the molecular movements, even in a few picoseconds. Therefore, this is a much shorter time then is generally of interest, but useful information can be gained from short simulations.

Molecular dynamics mimics the way a molecule actually explores its conformational space, rather than trying to get a picture of the whole of the conformational space, as conformational searching methods do. This is an advantage if the conformational space is so large or so intricate that conformational search methods cannot be continued until a complete

picture begins to emerge. Molecular dynamics, therefore, is particularly suited to studying protein conformations, or other large molecules for which there is incomplete structural data.

D. Penalty Methods

In mathematical terms, a penalty function is a function $P(x)$ which is chosen such that the optimal solution of the constrained problem:

$$\text{minimize } z = f(x) \qquad (2.26)$$

subject to one or more constraints on x, is close to the optimal solution of the unconstrained problem:

$$\text{minimize } Z = f(x) + \gamma\, P(x) \qquad (2.27)$$

where the constant γ is called the *penalty parameter*, and the function P is called the *penalty function*. [85]

In other words, the penalty function models a constrained optimization problem as an unconstrained problem. The solution obtained from this unconstrained problem is then used to solve the original problem. Of course, the solution to the unconstrained problem (the approximated solution) may not be exactly equal to the solution to the constrained problem (the true solution). Whether or not the solution to the unconstrained problem is a good approximation to the true solution depends on the penalty parameter γ and the penalty function P. We would expect that the larger the value of the penalty parameter γ, the closer the approximated solution will be to the true solution, since points that violate the constraints are penalized more heavily. Ideally, in the limit as $\gamma \rightarrow \infty$, the penalty method should yield the true solution to the constrained problems.

REFERENCES

1. Allinger, N. L., Conformational analysis. 130. MM2. A hydrocarbon force field utilizing V1 and V2 torsional terms. *Journal of the American Chemical Society* **1977,** 99, (25), 8127 - 8134.
2. *Chemical Graph Theory*. CRC Press: Boca Raton, 1983.
3. Beck, A.; Bleicher, M.; Crowe, D., *Excursion into Mathematics*. Worth: 1969.
4. Volarath, P.; Wang, H.; Fu, H.; Harrison, R. In *Knowledge-Based algorithms for chemical structure and property analysis*, EMBS 26th IEEE EMBS Annual International Conference, California, USA, 2004; California, USA, 2004.
5. Engel, T., *Chemoinformatics*. John Wiley & Sons: Germany, 2003.
6. Dalby, A.; Nourse, J. G.; Hounshell, W. D.; Gushurst, A. K. I.; Grier, D. L.; Leland, B. A.; Laufer, J., Description of several chemical structure file formats used by computer programs developed at Molecular Design Limited *J. Chem. Inf. Comput. Sci.* **1992,** 32, 244-255.
7. Leach, A. R.; Gillet, V. J., *An Introduction to Chemoinformatics*. Springer: Netherlands, 2003.
8. Ganesan, N.; Bennett, N. F.; Velauthapillai, M.; Pattabiraman, N.; Squier, R.; Kalyanasundaram, B., Web-based interface facilitating sequence-to-structure analysis of BLAST alignment reports. *Biotechniques* **2005,** 39, (2), 186, 188.
9. Margelevicius, M.; Venclovas, C., PSI-BLAST-ISS: an intermediate sequence search tool for estimation of the position-specific alignment reliability. *BMC Bioinformatics* **2005,** 6, 185.
10. Labesse, G., MulBlast 1.0: a multiple alignment of BLAST output to boost protein sequence similarity analysis. *Comput Appl Biosci* **1996,** 12, (6), 463-7.
11. Wiswesser, W. J., *A Chemical Line-Formula Notation*. Crowell Co.: New York, 1954.
12. Weininger, D., SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules *J. Chem. Inf. Comput. Sci.* **1988,** 28, 31-36.
13. Weininger, D.; Weininger, A.; Weininger, J. L., SMILES. 2. Algorithm for generation of unique SMILES notation. *J. Chem. Inf. Comput. Sci.* **1989,** 29, 97-101.
14. http://www.daylight.com/dayhtml/doc/theory/theory.smiles.html.
15. http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html.
16. http://www.daylight.com/meetings/summerschool01/course/basics/smirks.html.
17. Holde, K. E. v.; Johnson, W. C.; Ho, P. S., *Principles of Physical Biochemistry*. Prentice-Hall: Englewood Cliffs, NJ, 1998.
18. Leach, A. R., *Molecular Modelling: Principles and Applications*. 2nd ed.; Prentice Hall: Great Britain, 2001.
19. Rigby, M.; Smith, E. B.; Wakeham, W. A.; Maitland, G. C., *The Forces between Molecules*. Clarendon Press: Oxford, 1986.
20. Halgren, T. A., The representation of van der Waals (vdW) interactions in molecular mechanics force fields: potential form, combination rules, and vdW parameters. **1992,** 114, (20), 7827-7843.
21. Halgren, T. A., Merck molecular force field. I. Basis, form, scope, parameterization, and performance of MMFF94. *Journal of Computational Chemistry* **1996,** 17, (5-6), 490-519.
22. Halgren, T. A., Merck molecular force field. II. MMFF94 van der Waals and electrostatic parameters for intermolecular interactions. *Journal of Computational Chemistry* **1996,** 17, (5-6), 520-552.

23. Buckingham, A. D., Molecular Quadrupole Moments. *Chemical Society (Great Britain), Quarterly Reviews* **1959,** 13, 183-214.
24. Hill, T. L., Physical Interaction of Electrons with Dielectric Media. *J. Chem. Phys.* **1948,** 16, 394.
25. Vedani, A., YETI: An interactive molecular mechanics program for small-molecule protein complexes. *Journal of Computational Chemistry* **1988,** 9, (3), 269-280.
26. Goodford, P. J., A computational procedure for determining energetically favorable binding sites on biologically important macromolecules. **1985,** 28, (7), 849-857.
27. Maple, J. R.; Dinur, U.; Hagler, A. T., Derivation of Force Fields for Molecular Mechanics and Dynamics from ab initio Energy Surfaces. *PNAS* **1988,** 85, (15), 5350-5354.
28. Dinur, U.; Hagler, A. T., *Reviews in Computational Chemistry*. VCH: New York, 1991; Vol. 2, p 99.
29. Hwang, M. J.; Stockfisch, T. P.; Hagler, A. T., Derivation of Class II Force Fields. 2. Derivation and Characterization of a Class II Force Field, CFF93, for the Alkyl Functional Group and Alkane Molecules. **1994,** 116, (6), 2515-2525.
30. Allinger, N. L.; Chen, K.; Katzenellenbogen, J. A.; Wilson, S. R.; Anstead, G. M., Hyperconjugative effects on carbon - Carbon bond lengths in molecular mechanics (MM4). *Journal of Computational Chemistry* **1996,** 17, (5-6), 747-755.
31. Allinger, N. L.; Yuh, Y. H.; Lii, J. H., Molecular mechanics. The MM3 force field for hydrocarbons. 1. **1989,** 111, (23), 8551-8566.
32. Allinger, N. L.; Li, F.; Yan, L., Molecular mechanics. The MM3 force field for alkenes. *Journal of Computational Chemistry* **1990,** 11, (7), 848-867.
33. Allinger, N. L.; Chen, K.; Lii, J.-H., An improved force field (MM4) for saturated hydrocarbons. *Journal of Computational Chemistry* **1996,** 17, (5-6), 642-668.
34. Zhou, X.; Allinger, N. L., Molecular mechanics calculations (MM3) on alkyl iodides. *Journal of Physical Organic Chemistry* **1994,** 7, (8), 420-430.
35. Bagossi, P.; Zahuczky, G.; Tözsér, J.; Weber, I. T.; Harrison, R. W., Improved Parameters for Generating Partial Charges: Correlation with Observed Dipole Moments. *Journal of Molecular Modeling* **1999,** 5, (9), 143-152.
36. Weber, I. T.; Harrison, R. W., Molecular mechanics calculations on HIV-1 protease with peptide substrates correlate with experimental data. *Protein Eng.* **1996,** 9, 679-690.
37. Bagossi, P.; Zahuczky1, G.; Tözsér, J.; Weber, I. T.; Harrison, R. W., Improved Parameters for Generating Partial Charges: Correlation with Observed Dipole Moments. *Journal of Molecular Modeling* **1999,** 5, (9), 143-152.
38. Beachy, M. D.; Chasman, D.; Murphy, R. B.; Halgren, T. A.; Friesner, R. A., Accurate ab Initio Quantum Chemical Determination of the Relative Energetics of Peptide Conformations and Assessment of Empirical Force Fields. **1997,** 119, (25), 5908-5920.
39. Schlecht, M. F., *Molecular Modeling on the PC*. Wiley-VCH: New York, 1998.
40. Wertz, D. H.; Allinger, N. L., *Tetrahedron* **1974,** 30, (1579).
41. Allinger, N. L.; Hindman, D.; Hoenig, H., Conformational analysis. 125. The importance of twofold barriers in saturated molecules. **1977,** 99, (10), 3282-3284.
42. Allinger, N. L.; Chen, K.; Rahman, M.; Pathiaseril, A., Molecular mechanics (MM3) calculations on aldehydes and ketones. **1991,** 113, (12), 4505-4517.
43. Lii, J. H.; Allinger, N. L., Molecular mechanics. The MM3 force field for hydrocarbons. 2. Vibrational frequencies and thermodynamics. **1989,** 111, (23), 8566-8575.

44. Allinger, N. L.; Chen, K.; Lii, J.-H., An improved force field (MM4) for saturated hydrocarbons. *Journal of Computational Chemistry* **1996,** 17, 642-668.
45. Nevins, N.; Chen, K.; Allinger, N. L., Molecular mechanics (MM4) calculations on alkenes. *Journal of Computational Chemistry* **1996,** 17, 669-694.
46. Nevins, N.; Allinger, N. L., Molecular mechanics (MM4) vibrational frequency calculations for alkenes and conjugated hydrocarbons. *Journal of Computational Chemistry* **1996,** 17, 730-746.
47. Weiner, P. K.; Kollman, P. A., AMBER: Assisted model building with energy refinement. A general program for modeling molecules and their interactions. *Journal of Computational Chemistry* **1981,** 2, (3), 287-303.
48. Weiner, S. J.; Kollman, P. A.; Case, D. A.; Singh, U. C.; Ghio, C.; Alagona, G.; Profeta, S.; Weiner, P., A new force field for molecular mechanical simulation of nucleic acids and proteins. **1984,** 106, (3), 765-784.
49. Weiner, S. J.; Kollman, P. A.; Nguyen, D. T.; Case, D. A., An all atom force field for simulations of proteins and nucleic acids. *Journal of Computational Chemistry* **1986,** 7, (2), 230-252.
50. Brooks, B. R.; Bruccoleri, R. E.; Olafson, B. D.; States, D. J.; Swaminathan, S.; Karplus, M., CHARMM: A program for macromolecular energy, minimization, and dynamics calculations. *Journal of Computational Chemistry* **1983,** 4, (2), 187-217.
51. Nilsson, L.; Karplus, M., Empirical energy functions for energy minimization and dynamics of nucleic acids. *Journal of Computational Chemistry* **1986,** 7, (5), 591-616.
52. Smith, J. C.; Karplus, M., Empirical force field study of geometries and conformational transitions of some organic molecules. *J. Am. Chem. Soc.* **1992,** 114, (3), 801-812.
53. Schmidt, J. M.; Brueschweiler, R.; Ernst, R. R.; Dunbrack, R. L.; Joseph, D.; Karplus, M., Molecular dynamics simulation of the proline conformational equilibrium and dynamics in antamanide using the CHARMM force field. *J. Am. Chem. Soc.* **1993,** 115, (19), 8747-8756.
54. MacKerell, A. D.; Wiorkiewicz-Kuczera, J.; Karplus, M., An all-atom empirical energy function for the simulation of nucleic acids. *J. Am. Chem. Soc.* **1995,** 117, (48), 11946-11975.
55. Blondel, A.; Karplus, M., New formulation for derivatives of torsion angles and improper torsion angles in molecular mechanics: Elimination of singularities. *Journal of Computational Chemistry* **1996,** 17, (9), 1132-1141.
56. Rappe, A. K.; Casewit, C. J. R., *Molecular Mechanics Across Chemistry*. University Science Books: Fort Collins, CO, 1997.
57. Rappe, A. K.; Casewit, C. J.; Colwell, K. S.; Goddard, W. A.; Skiff, W. M. I., UFF, a full periodic table force field for molecular mechanics and molecular dynamics simulations. *J. Am. Chem. Soc.* **1992,** 114, (25), 10024-10035.
58. Casewit, C. J.; Colwell, K. S.; Rappe, A. K., Application of a universal force field to organic molecules. *J. Am. Chem. Soc.* **1992,** 114, (25), 10035-10046.
59. Casewit, C. J.; Colwell, K. S.; Rappe, A. K., Application of a universal force field to main group compounds. *J. Am. Chem. Soc.* **1992,** 114, (25), 10046-10053.
60. Badger, R. M., A Relation Between Internuclear Distances and Bond Force Constants. *J. Chem. Phys.* **1934,** 2, (3), 128-131.
61. Badger, R. M., The Relation Between the Internuclear Distances and Force Constants of Molecules and Its Application to Polyatomic Molecules. *J. Chem. Phys.* **1935,** 3, (11), 710-714.

62. Herschbach, D. R.; Laurie, V. W., Anharmonic Potential Constants and Their Dependence upon Bond Length. *J. Chem. Phys.* **1961,** 35, (2), 458-463.
63. Halgren, T. A., Maximally diagonal force constants in dependent angle-bending coordinates. II. Implications for the design of empirical force fields. *J. Am. Chem. Soc.* **1990,** 112, (12), 4710-4723.
64. Mu, Y. G.; Kosov, D. S.; Stock, G., Conformational Dynamics of Trialanine in Water. 2. Comparison of AMBER, CHARMM, GROMOS, and OPLS Force Fields to NMR and Infrared Experiments. *J. Phys. Chem. B.* **2003,** 107, (21), 5064-5073.
65. Woutersen, S.; Hamm, P., Structure Determination of Trialanine in Water Using Polarization Sensitive Two-Dimensional Vibrational Spectroscopy. *J. Phys. Chem. B.* **2000,** 104, (47), 11316-11320.
66. Woutersendagger, S.; Mu, Y.; Stock, G.; Hammdagger, P., Subpicosecond conformational dynamics of small peptides probed by two-dimensional vibrational spectroscopy. *Proc. Natl. Acad. Sci. U.S.A.* **2001,** 98, (20), 11254-11258.
67. Poon, C.-D.; Samulski, E. T.; Weise, C. F.; Weisshaar, J. C., Do Bridging Water Molecules Dictate the Structure of a Model Dipeptide in Aqueous Solution? *J. Am. Chem. Soc.* **2000,** 122, (23), 5642-5643.
68. Woutersen, S.; Pfister, R.; Hamm, P., Peptide conformational heterogeneity revealed from nonlinear vibrational spectroscopy and molecular-dynamics simulations. *J. Chem. Phys.* **2002,** 117, (14), 6833-6840.
69. Schweitzer-Stenner, R.; Eker, F.; Huang, Q.; Griebenow, K., Dihedral Angles of Trialanine in D2O Determined by Combining FTIR and Polarized Visible Raman Spectroscopy. *J. Am. Chem. Soc.* **2001,** 123, (39), 9628-9633.
70. Schweitzer-Stenner, R., Dihedral Angles of Tripeptides in Solution Directly Determined by Polarized Raman and FTIR Spectroscopy. *Biophys. J.* **2002,** 83, (1), 523-532.
71. Reddy, S. Y.; Leclerc, F.; Karplus, M., DNA Polymorphism: A Comparison of Force Fields for Nucleic Acids. *Biophys. J.* **2003,** 84, 1421-1449.
72. Wiberg, K. B., A Scheme for Strain Energy Minimization. Application to the Cycloalkanes1. *J. Am. Chem. Soc.* **1965,** 87, (5), 1070-1078.
73. Engler, E. M.; Andose, J. D.; Schleyer, P. V. R., Critical evaluation of molecular mechanics. *J. Am. Chem. Soc.* **1973,** 95, (24), 8005-8025.
74. Fletcher, R., *Practical Methods of Optimization*. Wiley & Sons: UK, 1990.
75. Boyd, R. H., Method for Calculation of the Conformation of Minimum Potential-Energy and Thermodynamic Functions of Molecules from Empirical Valence-Force Potentials—Application to the Cyclophanes. *Journal of Chemical Physics* **1968,** 49, (6), 2574-2583.
76. Altona, C.; Faber, D. H., Empirical force field calculations A tool in structural organic chemistry. In *Topics in Current Chemistry*, Springer: Berlin/Heidelberg, 1974; Vol. 45, pp 1-38.
77. Papalambros, P. Y.; Wilde, D. J., *Principles of Optimal Design: Modeling and Computation (2nd Edition)*. Cambridge University Press: USA, 2000.
78. Goodman, J., *Chemical Applications of Molecular Modeling*. Royal Society of Chemistry: Wiltshire, 1998.
79. Kirkpatrick, S.; Gelatt, C. D. J.; Vecchi, M. P., Optimization by Simulated Annealing. *Sciences* **1983,** 220, (4598), 671-680.
80. Holland, J. H., *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*. University of Michigan Press: Ann Arbor, MI, 1975.

81. McGarrah, D. B.; Judson, R. S., Analysis of the genetic algorithm method of molecular conformation determination. *Journal of Computational Chemistry* **1993,** 14, (11), 1385-1395.
82. Herrmann, F.; Suhai, S., Energy minimization of peptide analogues using genetic algorithms. *Journal of Computational Chemistry* **1995,** 16, (11), 1434-1444.
83. Meza, J. C.; Judson, R. S.; Faulkner, T. R.; Treasurywala, A. M., A comparison of a direct search method and a genetic algorithm for conformational searching. *Journal of Computational Chemistry* **1996,** 17, (9), 1142-1151.
84. Nair, N.; Goodman, J. M., Genetic Algorithms in Conformational Analysis. *J. Chem. Inf. Comput. Sci.* **1998,** 38, (2), 317-320.
85. Chong, E. K. P.; Zak, S. H., *An Introduction to Optimization*. John Wiley & Sons: Canada, 1996.

Chapter 3: Knowledge-based algorithms for chemical structure and property analysis

Designing molecules with appropriate chemical properties is an essential key in drug design [1, 2]. With the currently available technologies, chemical structures of small molecules can be easily created using computer programs, such as ChemDraw [3] and ISIS [4]. A more challenging task, therefore, lies in the next level: the analysis of chemical properties from these created structures. Accurate prediction of chemical properties using computer technology has many useful applications in the field of biotechnology, particularly in synthetic chemistry. There are currently a few programs that have been designed to aid in such a task. These programs rely heavily on the use of knowledge-based embedded databases [5, 6] and structure search algorithms [7-9]. However, although these programs are successful in their assigned tasks, there are still rooms in the field to be explored and improved [10-14].

In order to develop an automated approach to molecular design, it is necessary to develop 'rule-based' algorithms that proficiently analyze chemical properties of small molecules that are useful in the synthetic pathways prediction. These rule-based algorithms will form the basis for computational drug design algorithms.

3.1 Methodology

The source of the input files is the National Cancer Institute (NCI) [15] small molecule database (~120,000 compounds). All molecule files are in 2D .mol format generated by MACCS-II FROM NCI/DIS. Each file is identified via a unique CAS identification number. Twenty two molecule files are randomly selected as test molecules for the structure-based

chemistry determination algorithm. After testing all 600,000 molecules were converted to 3-D coordinate sets.

The program compiler used for the chemical property prediction algorithm is Microsoft Visual C++ version 6.0 and the Window operating system used is Window XP Professional. The Cartesian coordinates of the molecules are obtained from AMMP (Advanced Molecular Modeling Program) [16, 17], after the energy minimization. The algorithms for structure search are programmed using JAVA programming language (J2SE 1.4.2 SDK). The compiler is obtained from Sun (http://www.java.sun.com).

3.2 Results and Discussion

3.2.1 Structural-Based Chemistry Determination

The developed algorithm showed successful chemical property prediction on all of the twenty two test molecules, which are varied in structures and sizes, and then performed successfully on the entire data set. The approach used to design this algorithm involves two steps. The first step is the assignment of an appropriate atom type to each of the atoms, and the second is the analysis of the chemical properties from the assigned atom types.

To assign an appropriate atom type to each atom, a simple molecular tree is constructed from the connectivity table of the input file. Each node of the tree represents an atom, and a branch between any two nodes represents the bond order between the atoms. Since 2D .mol files do not contain hydrogen atoms, the root of the tree is simply the first atom on the list. The rest of the atoms are then connected to the root in the usual procedure until all atoms are connected.

In AMMP, atom types [18] are defined based on both the atoms' geometric arrangements in relation to their neighbors and their chemical properties. For instance, sp3 and sp1 carbons can

---

This work was partially supported by NIH1 R01 GM62920-04A1, NIH1 P20 GM065762-01A1, the Georgia Research Alliance and the Georgia Cancer Coalition

be easily distinguished and assigned as 'C3' and 'C1' atom types, respectively, due to their obvious geometry difference. However, sp2 carbons can be either a carbonyl carbon ('C2') involved in groups, such as ketones or aldehydes, or an aromatic carbon ('ca'). Hence, the chemical property of the carbon must first be examined before an appropriate atom type can be assigned. The full list of pre-defined atom types can be found in the AMMP help file [16, 18].

Once an appropriate atom type is assigned to all of the atoms, the next step is the derivation of the functional groups and the reactive sites within the molecules using logic-based approach. The reactive sites are derived from the polarity (the resultant electro-negativities) among the involved atoms. All regions that consist of atoms that have large electro-negativity differences (regardless of the types of atoms involved) are considered potential reaction sites. The geometry around the reactive regions is also evaluated based on the Cartesian coordinates obtained from the molecular model built with AMMP.

The procedure above can quickly determined the chemistry of simple functional groups, such as carboxylic acid derivatives, and amine containing groups. A more challenging task is the analysis of larger and more complex functional groups, such as fused rings. Fused aromatic rings are especially problematic because it is necessary to distinguish between ring systems that have double bonds and aromatic systems where all the bonds are equivalent. Since aromatic systems are often found in molecules, and aromatic chemistry is significantly different from non-aromatic chemistry, it is critical to correctly identify aromatic systems. Our approach to the aromatic ring identification is to derive an aromatic tree from the previously constructed tree.

The atom type of the aromatic carbon atoms is 'ca'. Other atoms such as nitrogen and oxygen may be present as well. Similarly, individual carbon atoms that were miss-identified as sp2 carbons could be part of an aromatic system. Since the locations and the connectivities of these atoms are known, the structure of the aromatic-tree can be easily derived. This algorithm is

designed to deal with both the best-case (when a molecule contains one ring or isolated rings), as well as the worse-case (when a molecule contains fused rings, like anthracenes, etc.) scenarios. In all cases, each aromatic atom is classified into one of two types: one is the *peripheral* aromatic atom, and the other is the *fused* aromatic atom. The peripheral aromatic atoms are those atoms that have two aromatic neighbors. The fused aromatic atoms are the atoms that have three aromatic neighbors, which is an indication that the atom is at the junction between two rings. A peripheral aromatic atom is randomly selected to be the root of the tree. Once the root is established, two of its aromatic neighbors are then traced simultaneously. In a best case scenario where there is only one ring or a number of isolated rings, all aromatic atoms are classified as peripheral atoms. In the fused rings situation, both aromatic atom types exist. The worse-case scenario for fused ring situations is when each of the fused atoms has two aromatic neighbors.

To handle this problem, both of the first generation children are used. For example, consider a ring with a peripheral aromatic root ***R***. ***R*** has two children ***A*** and ***B***, each with two children, ***1*** and ***2***. In order to collect the members of a ring, all combinations of the aromatic grandchildren are evaluated. If any of the grandchildren has two different parents, then the atoms involved in the sub-tree belong to the same ring. For each ring, after all of its six members are determined, only those members with two aromatic neighbors are eliminated from the list. The members with three aromatic neighbors are at the fused point of two rings and, therefore, allowed to remain in the list. This design is to ensure that all rings will have six members, although some of the members may belong to more than one ring. This process is repeated until all aromatic atoms are assigned to a ring. As mentioned previously that, the neighbors of each atom have been defined in the atom-type tree. Therefore, in addition to accurately constructing complex aromatic rings, this algorithm can also identify the substituents at the meta-, ortho-, and para-positions of the rings.

3.2.2 Structural Search Algorithm

With a database containing approximately 120,000 compounds, it is necessary to design an algorithm that can quickly search for the compounds of interests. We have developed a simple rule-based algorithm that is specifically designed for 'quick' exact- and sub- structure searching. Various sub-structures (carboxylic derivatives, aromatic rings, and amine containing compounds) have been used to validate the performance of the algorithms. In all cases, thus far, the algorithm has successfully returned the compounds that contain the desired sub-structures. The algorithm is currently under the continuous testing to ensure its accuracy.

In order to represent a molecule, the algorithm first constructs the molecule using an *adjacency matrix*. The elements of both row and the column are the atoms listed in the input molecular file. The elements in the cells are the bond order between two atoms. The next stage is the designing of the molecular tree and the search algorithm. The construction of the tree is further divided into two sub-steps: choosing a unique root for the tree and building the rest of the tree from the chosen root.

A. Choosing a unique root and 'best-tree' algorithm

Choosing an appropriate root is critical. The criterion for choosing a root is that the resultant tree must have a unique one-to-one relationship to the molecule. The rules that we use as the criteria of choosing the unique root is as followed:

A. Atom symbol: select the shorter length. For example, 'C' (carbon) is shorter than 'Na'. Hence, 'C' is selected over 'Na'.

B. If 'A' outputs more than one root, then select the root with the lowest IUPAC (International Union of Pure and Applied Chemists) codes. For example, 'C' is lower in IUPAC coding order than 'O' (oxygen). Hence, 'C' is selected over 'O'.

C. If 'B' outputs more than one root, select the root with the smallest number of connected neighbors.

D. If 'C' outputs more than one root, select the root with the smallest number of children.

E. If 'D' outputs more than one root, select the root with the smallest number of grand-children

The outlined rules basically represent a minimum spanning tree (MST) [19] that uses the number of children and grandchildren as the distance between atoms. For aromatic systems, since a tree does not contain a circle, a pseudo-node is added to the tree to complete the ring. Each of the potential roots is tested systematically using the above rules, until only one root is left in the list. However, it is possible that the above rules may still generate more than one root. If this is the case, then the 'best' tree ('best-tree' algorithm) must be selected based on the following criteria:

A. The tree that has the highest number of generations.

B. If 'A' outputs more than one tree, then the tree that has the smallest number of children is selected.

C. If 'B' outputs more than one tree, then the tree that has the smallest number of grand-children is selected.

This algorithm can output more than one tree if there is symmetry in the molecule. However, if there is symmetry in the molecule, then by definition the atoms are equivalent and it does not matter which tree is chosen. In this case we arbitrarily choose the first tree.

B. Building a molecular tree

Once the root of the tree is chosen, building the rest of the tree is a straightforward process. However, a care must be taken if the molecule of interest contains aromatic rings. For

compounds with aromatic rings, the atoms that are involved in the systems and those are not are separated. This is to ensure that the root is not an aromatic atom, unless the compound contains only aromatic rings. If this is the case, the root of the aromatic compound is determined by using the 'best' tree algorithm. The rest of the tree is then built using the connectivity obtained from the adjacent matrix. This process continues until all atoms in the molecule are connected.

C. Exact- and Sub- Structure Searching

Exact structure search can be performed in a straightforward fashion. First of all, the trees of the target (Structure A -- the desired sub-structure) and the comparing (Structure B – the molecule to be searched) molecules are built using the procedures outlined previously. The atoms within these trees are compared using depth-first search (DFS) algorithm [20]. In DFS, each node in both trees is compared. If both nodes are identical, the search continues to their first child in the next generation. In the exact-structure searching, all atoms from both trees are identical, and, therefore, it is the best-case. However, in the sub-structure searching, miss-mated atoms are expected. When a miss-match is encountered, the algorithm returns to the roots of the trees. The previously stored nodes from Structure A are deleted, and the same search procedure is repeated with the second child in the first generation of Structure B. This process continues until all atoms in Structure A are found in Structure B. This indicates that Structure A is a sub-structure of Structure B. Otherwise, Structure B does not contain such a sub-structure.

3.3 Conclusion

We have successfully developed two groups of algorithms: one is for an efficient sub- and exact-structure searching, and the other is for accurate chemical properties description of small molecules. This project is about chemical design. Hence, the search algorithms will be useful in the situations where parts of the desired structure are known. In such a situation, the database can be searched for fragments that share similarity to the molecular parts. These fragments can then

be tested, by combining them based on chemical rules, to determine if any of the combined structures satisfies the criteria. As for the chemical property prediction algorithm, it can be use in several situations. For instance, the algorithm can be implemented as a part of the 2D to 3D structure conversion. It can also be used as search criteria for the fragments in the database. Finally, it can be used to determine the chemical functional groups that are embedded in the 2D structures, which is discussed in Chapter 4.

REFERENCES

1. Clark, D. E.; Pickett, S. D., Computational Methods for the Prediction of 'Drug-likeness'. *Drug Discovery Today* **2000,** 5, 49-58.
2. Durham, S. K.; Pearl, G. M., Computational Methods to Predict Drug Liabilities *Current Opinion in Drug Discovery and Development* **2001,** 4, 110-115.
3. ChemDraw CambridgeSoft Corporation. http://www.cambridgesoft.com/
4. MDL®ISIS Draw MDL Information System http://www.mdli.com/
5. Edgar, S.; Holliday, J. D.; Willett, P., Effectiveness of Retrieval in Similarity Searches of Chemical Databases: A Review of Performance Measures *Journal of Molecular Graphics and Modelling* **2000,** 18, 343-357.
6. Hagadone, T. R., Molecular Substructure Similarity Searching: Efficient Retrieval in Two-Dimensional Structure Databases *Journal of Chemical Information and Computer Sciences* **1992,** 32, 515-521.
7. Downs, G. M.; Willett, P.; Fisanick, W., Similarity Searching and Clustering of Chemical Structure Databases Using Molecular Property Data *Journal of Chemical Information and Computer Sciences* **1994,** 34 1094-1102.
8. Willett, P.; Bernard, J. M.; Downs, G. M., Chemical Similarity Searching *Journal of Chemical Innformation and Computer Sciences* **1998,** 38, 983-996.
9. Burden, F. R., Molecular Identification Number of Substructure Searches *Journal of Chemical Information and Computer Sciences* **1989,** 29, 225-227.
10. Leach, A. R.; Gillet, V. J., *An Introduction to Chemoinformatics*. Kluwer Academic Publishers: Boston, MA, 2003.
11. Barnard, J. M., Problems of Substructure Search and Their Solution. In *Chemical Structures. The International Language of Chemistry*, Warr, W. A., Ed. Springer-Verlag: Berlin, 1988; pp 113-126.
12. Lowell, H.; Hall, L. H.; Kier, L. B., Issues in Representation of Molecular Structure: The Development of Molecular Connectivity *Journal of Molecular Graphics and Modeling* **2001,** 20, 4-18.
13. Barnard, J. M., Substructure Searching Methods: Old and New. *Journal of Chemical Information and Computer Sciences* **1993,** 33, 532-538.
14. Livingstone, D. J., The Characterization of Chemical Structures Using Molecular Properties. A Survey. *Journal of Molecular Graphics and Modelling* **2000,** 40, 195-209.
15. NCI. National Cancer Institute. http://www.nci.nih.gov

16. Harrison, R. W. Advanced Molecular Modeling Program (AMMP). http://www.cs.gsu.edu/~cscrwh/
17. Harrison, R. W.; Chatterjee, C.; Weber, I. T., Analysis of six protein structures predicted by comparative modeling techniques *Proteins: Structure Function and Genetics* **1995,** 23, 463-471.
18. Bagossi, P.; Tozser, J.; Weber, I. T.; Harrison, R. W., Modification of parameters of the charge equilibrium scheme to achieve better correlation with experimental dipole moments. *J. Mol. Model.* **1998,** 5 143-152.
19. Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C., In *Introduction to Algorithms 2nd ed.*, The MIT Press London, England, 2001; pp 561-579.
20. Cormen, T. H.; Leiserson, C. E.; Rivest, R. L.; Stein, C., In *Introduction to Algorithms, 2nd ed.*, The MIT Press: London, England, 2001; pp 541-547.

Chapter 4: Program Designs for Chemical Functional Groups and their Reactivity

A chemical functional group is a collection of atoms and bond orders (single, double, and triple bonds) that form a specific connectivity pattern, Figure 4.1.

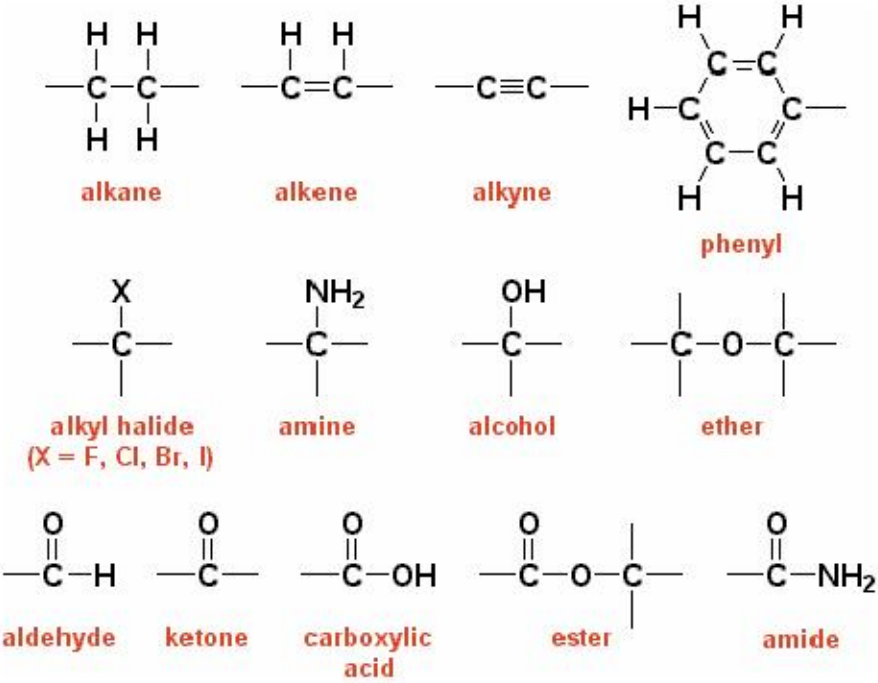


Figure 4.1: Examples of commonly found functional groups in organic chemistry [1]

The knowledge of the functional groups in a molecule is important because the syntheses of the subsequent compounds are planned based on their chemistry. [2, 3] Simple functional groups can be combined to form complex molecules, Figure 4.2.

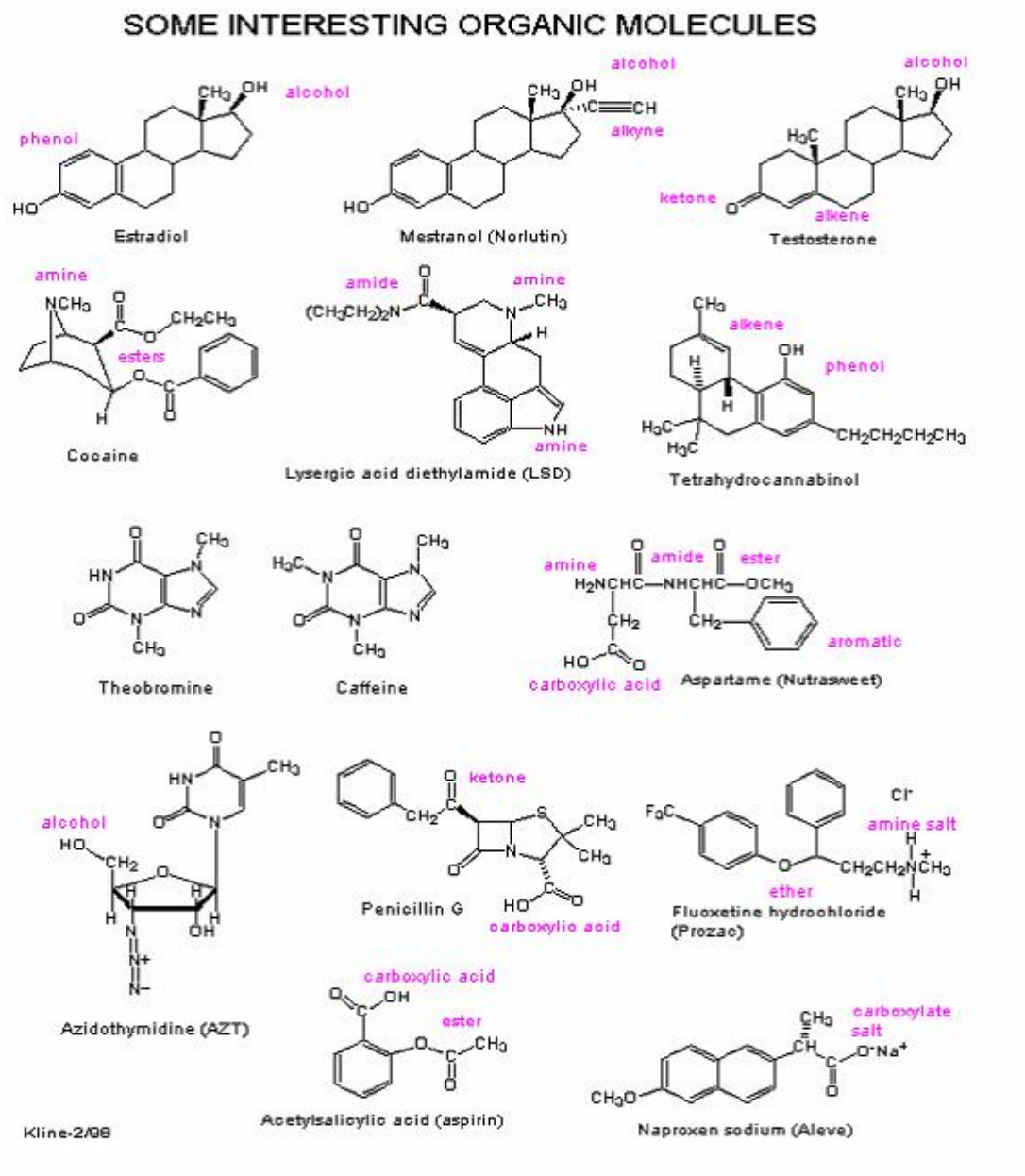


Figure 4.2: Molecules containing several functional groups [4]

As the molecule becomes more complex, the presences of the functional groups in the molecule no longer guarantee their reactivity. Reactivity of functional groups is environmentally sensitive. A slight change in the pH, for example, can immediately affect the chemistry of the functional groups and alter the behavior of the molecule. For these reasons, in order to predict

the chemical reactions, a set of programs that search for the functional group connectivity patterns and determine their reactivity is created.

4.1 Program Designs

4.1.1 Functional Group Search Programs (FGSP)

Thirteen test functional groups are selected for this work: amide, benzene, acid, ester, amine, alcohol, ether, alkene, alkyne, thiol nitrile, ketone, and aldehyde.

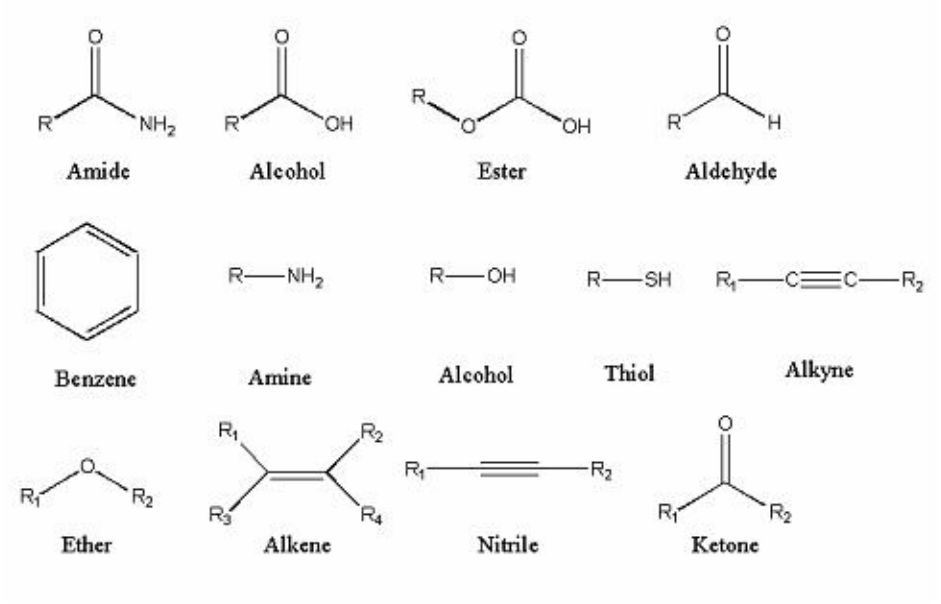


Figure 4.3: The thirteen test functional groups for this work

A search program is created for each of the functional groups. The design of the search programs is described in Chapter 3. Each of the search programs is designed to seek for *a signature pattern*. A functional group signature pattern refers to the atom types and their connectivity that are responsible for the functional group's chemistry. For instance, the signature pattern in an amide, Figure 4.4, is shown below:

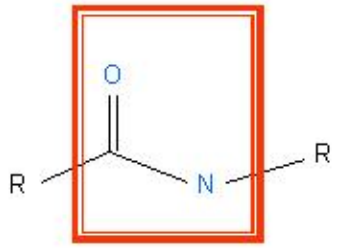


Figure 4.4: The signature pattern of the amide are enclosed in red

The search programs are applied individually to an input .mol file to locate their patterns in the molecule. The tables listed in Appendix A outline the search algorithmic procedures for each of the search programs. Each of the programs generates a .txt output file that contains the name of its functional group, the numbers and symbols of the functional group atom members, and the total number of active bonds (double and/or triple bonds) that are present in the functional group. The member atom numbers and symbols are exactly the same as in the .mol input file. This is to retain the connectivity information in the original molecule. The results in the output files from the thirteen search programs are combined into one .txt file. The combined results are then analyzed by the functional group scoring program.

4.1.2 Functional Group Scoring Program

The goal of the functional group scoring program is to determine the reactivity of the functional groups that are present in a molecule. One functional group is being evaluated at a time, and it is referred to in this work as a *target functional group* (TFG). The reactivity of the TFG is impacted by two factors: (1) the number of the TFG atoms that is being shared, and (2) the electronegativity affect from the neighbors.

A. The Atom Sharing Impact (ASI) on the TFG's Overall Reactivity

The atom sharing factor is the first to be considered in the TFG's reactivity because, if present, it has a direct impact on the chemistry of the TFG. Only the sharing with a neighbor functional group whose reactivity values (R.V.s) is higher than that of the TGF is considered.

This is because functional groups with higher R.V.s are assumed to be more stable and can destabilize the neighbors with lower R.V.s.

The R.V. of a functional group is determined using the following equation:

$$\text{R.V.} = \text{Average ENV} + \text{TNABs} \qquad (4.1)$$

where ENV = electronegativity value and TNABs = total number of active bonds

The average ENV of the functional group can be obtained using the following equation:

$$\text{Average ENV} = \frac{\sum ENV_{a1} + ENV_{a2} + ENV_{a3} + \ldots + ENV_{aN}}{N} \qquad (4.2)$$

where a = atomic element, and N = number of atoms present in the functional group

The ENVs used in this work are shown in Appendix B. For example, according to the electronegativity chart, the ENV for carbon is 2.55, for nitrogen is 3.04, for oxygen is 3.44, and so forth. The total number of active bonds (TNABs) refers to the total number of double and triple bonds that are present in the functional group. For examples, the TNABs in ester is 1 for the double bond between the carbonyl carbon and oxygen, and, likewise, the TNABs in nitrile is also 1 for the triple bond on the nitrogen.

Each of the pre-defined functional groups has a default average ENV and a default TNABs. If the TFG shares its atoms with a pre-defined functional group (PDFG), the PDFG's default average ENV and TNAB are used to calculate its R.V. The same method is applied to the calculation of the TFG's R.V. The significance of the R.V. is demonstrated in an example compound, Figure 4.5, in which the lesser reactive functional group amine is a part of a more reactive functional group amide.

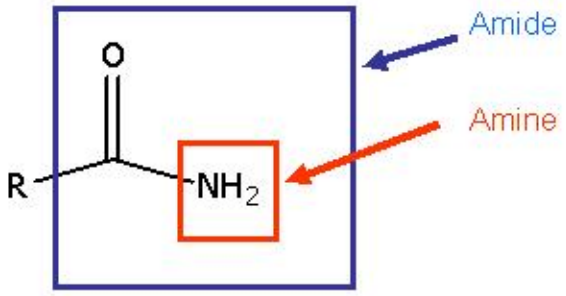


Figure 4.5: The amide atoms are enclosed in the blue box, and the amine atoms are enclosed in the red box

According to Equation 4.1, the amine has a R.V. of 3.04; while, the amide has a R.V. of 4.01. For the amine, the total number of the shared atoms is 1. This is equivalent to 100% reactivity loss for the amine because its only atom, the nitrogen, is being shared with the amide, which has a higher R.V. On the other hand, although the amide shares 1/3 of its atoms with the amine, this does not affect its reactivity because the amine has a lower R.V. Hence, the amide remains 100% active.

Once the R.V.s of all the PDFGs have been determined, they are individually compared to the R.V. of the TFG to determined the total number of higher R.V. neighbor functional groups. This number is used to determine the shared atom impact factor (SAIF) of the higher R.V. neighbor functional groups (higher R.V. NFGs) on the TFG:

$$\text{SAIF} = \left(\frac{SA}{TA}\right) \text{ x } 100 \qquad (4.3)$$

where SA = number of the TFG shared atoms with the higher R.V. neighbors, and TA = the total number of atoms present in the functional group.

The SAIF impact upon the overall TFG reactivity (OTFGR) is obtained using the following Equation 4.4:

$$\text{OTFGR} = 100 - \text{SAIF} \qquad (4.4)$$

The OTFGR value indicates the percent identity of the functional group remains after the SAF is subtracted. For a TFG that does not share its atoms with any functional group neighbor with a higher R.V., its SAIF is zero. Hence, its OTFGR value is 100, which indicates that the TFG remains 100% active.

B. Affects of Neighboring Atoms on the OTFGR

In this work, the neighbors surrounding the TFG are divided into three classes, based on the bond distances between them and the TFG. The neighbors that are one bond away from the TFG are referred to as the *first-layer neighbors*. Those that are two and three bonds away are referred to as the *second-layer neighbors* and the *third-layer neighbors*, respectively. The same classification is applied to the active bonds. The active bonds that are found within a bond away from the TFG are referred to as the *first-layer active bonds*. Those that are two and three bonds away are referred to as the *second-layer active bonds* and the *third-layer active bonds*, respectively. Once all the atoms and bonds are determined, appropriate functional groups are assigned to them.

Each of the layer neighbors is assigned a significant percentage. For the first-layer neighbors, a significant percentage of 99% is assigned. For the second-layer and the third-layer neighbors, the significant percentages of 66% and 33% are assigned, respectively. This significant percentage assignment is designed based on the fact that the influences of the neighbors on the TFG decrease as the distance between them increases. The neighbors and bonds that are located more than three bonds away from the TFG are assumed to have insignificant impact on the TFG reactivity.

The significant values (S.V.) of each of the layer neighbors are obtained using the following equations:

$$S.V._1 = [(\sum (\text{average ENV} + \text{AB})_{X,1}) * 99] \qquad (4.5)$$

$$S.V._2 = [(\sum (\text{average ENV} + \text{AB})_{X,2}) * 66] \qquad (4.6)$$

$$S.V._3 = [(\sum (\text{average ENV} + \text{AB})_{X,3}) * 33] \qquad (4.7)$$

where S.V. = significant value
1, 2, 3 = layer number
ENVs = electronegativity values
ABs = active bonds
X = functional group number

The atoms and the bonds are evaluated one layer at a time. In the first layer, only the functional group neighbors with higher ENVs are considered. This is because this layer is the closest to the TFG; hence, the atoms and active bonds in these layers can have a major effect on the chemistry of the TFG. For the second and the third layer, the average ENV and the active bonds of all functional groups in these layers are used. These two layers are farther away from the TFG. The impact of the functional groups in these layers is assumed to have lesser effect on the TFG. The obtained significant values are subtracted individually from the OTFGR. The equation that summarizes the overall scoring function is:

$$OTFGR = ((((100 - SAIF) - S.V._1) - S.V._2) - S.V._3) \qquad (4.8)$$

4.2 Results

4.2.1 Analysis of the Functional Group Search Programs

The search procedures for most of the functional groups can be performed straightforwardly. However, the procedures for the fused benzene rings require additional steps, due to the variations of the representations that can be generated by the ChemDraw Program. Figure 4.6 shows two possible representations of fused rings.

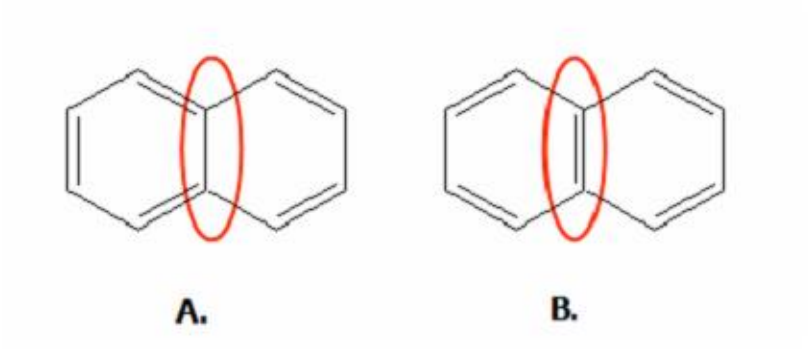


Figure 4.6: Two representations of benzene rings

As shown in Figure 4.3, the two equivalent fused rings can be represented in two different ways. Designing a program that uses the conventional requirements (3 double bonds, 3 single bonds, and 6 carbon atoms) for the benzene ring recognition is, therefore, insufficient. As shown in Figure 4.3A, if the conventional method was used, the second ring on the right would have been ignored by the program, even though it is aromatic. More ring representations can be found as the number of rings increases, Figure 4.7.

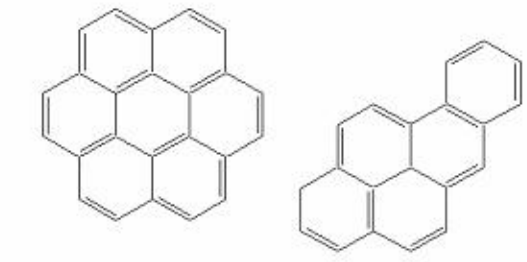


Figure 4.7: Different representations of benzene rings

The results show that the benzene search algorithm is capable of recognizing linear forms of fused benzene rings, Figure 4.8.

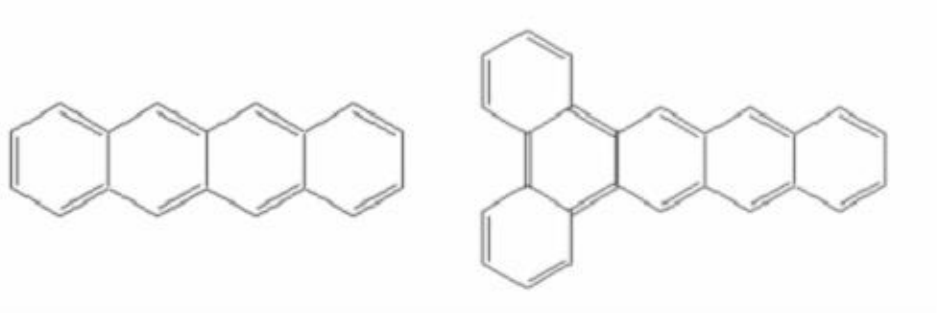


Figure 4.8: Linear ring representations that are recognizable by the program

However, with the non-linear ring forms, the program can only recognize up to five rings, Figure 4.9.

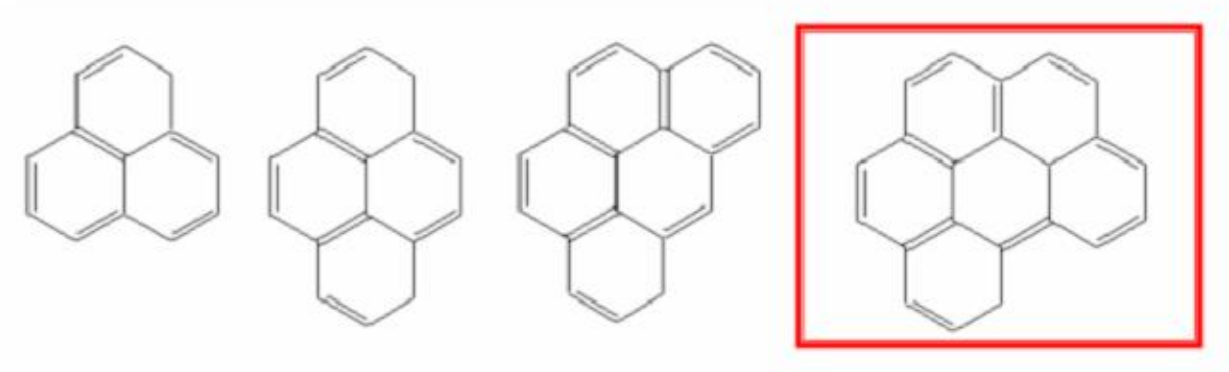


Figure 4.9: The non-linear ring forms that are recognizable by the algorithm. In the last structure, only the rings in blue are recognizable by the program

Branched rings have also been tested. One of the test branched structure is shown in Figure 4.10. All the rings are recognizable, except the one in red. The reason is due to the search logic being too restricted.

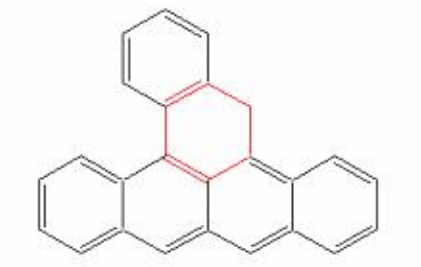


Figure 4.10: Test structure for branched rings

However, although there is a limitation in this program, the algorithm is still considered sufficient for the work. This is because the molecules that are used in the work are small organic molecules. Most of the structures comprises only of the recognizable ring forms, and the chances of encountering structures that have six or more non-linear ring forms is unlikely.

One of the test molecules (Molecule1) is shown in Figure 4.11.

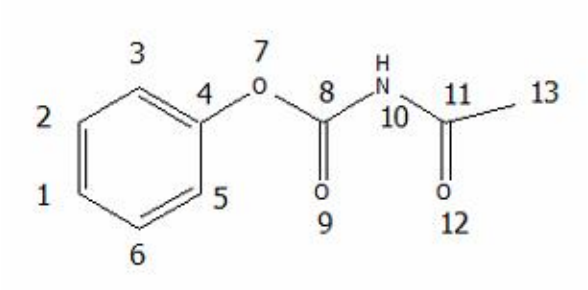


Figure 4.11: The structure of the test molecule, Molecule 1

The FGSPs are applied to the structure, and the combined result is shown below:

```
NAME Amide
ACTIVE_BOND 1
$$$$
NAME Amide
MEMBER 8 C
MEMBER 9 O
MEMBER 10 N
ACTIVE_BOND 1
$$$$
NAME Amide
MEMBER 10 N
MEMBER 11 C
MEMBER 12 O
ACTIVE_BOND 1
$$$$
NAME Benzen
MEMBER 1 C
MEMBER 2 C
MEMBER 6 C
MEMBER 4 C
MEMBER 3 C
MEMBER 5 C
ACTIVE_BONDS 6
$$$$
NAME Ester
MEMBER 9 O
MEMBER 8 C
MEMBER 7 O
ACTIVE_BONDS 1
$$$$
NAME Amine
MEMBER 10 N
ACTIVE_BONDS 0
$$$$
NAME Alcohol
ACTIVE_BONDS 0
$$$$
NAME Ether
MEMBER 7 O
ACTIVE_BONDS 0
$$$$
NAME Alkene
MEMBER 1 C
MEMBER 2 C
MEMBER 3 C
MEMBER 4 C
MEMBER 5 C
MEMBER 6 C
ACTIVE_BONDS 3
$$$$
NAME Alkyne
ACTIVE_BONDS 1
$$$$
NAME Thiol
ACTIVE_BONDS 0
$$$$
NAME Nitrile
ACTIVE_BONDS 1
$$$$
END
```

Figure 4.12: Combined output of test Molecule 1

In the result shown above, each of the functional groups is defined by its name (indicated by a keyword "NAME") followed by a list of their members (indicated by a keyword

"MEMBER"). The members are listed by their atom numbers, as appeared in the original .mol input file, followed by their atomic symbols. This ensures accurate atom identifications, which is critical for the subsequent scoring procedures. The keywords "$$$$" indicates the end of the functional group definition, and "END" indicates the end of the file. Finally, the keyword "ACTIVE_BONDS" indicates the total number of double bonds and triple bonds that are present in the molecule. Only the functional groups with a list of members are considered in the proceeding procedures. Those without members are ignored.

The results show that the programs are able to detect their target patterns in the input file. In addition, the programs could also associate accurately the identity of the functional groups with their atoms and assigned the correct number of the total active bonds. The purpose of this initial screening is to detect all functional group patterns that are present in the molecule, regardless of their reactivity. Therefore, the atom numbers 1, 2, 3, 4, 5, and 6 are detected as both benzene and alkene. The next step involves applying a scoring program to determine the actual reactivity of the functional groups.

4.2.2 Analysis of the Functional Group Scoring Function

The scoring program is a separate program that takes in both the original .mol structure file and the .txt file that contains a combined result of the functional group screening programs. The scoring program calculates the percent reactivity of the functional groups that have been previously determined. The scoring procedures are performed based on two criterions: (1) the total number of the atoms being shared in the TFG, and (2) the presence of the active atoms and bonds that are within the three-bond distance. The calculated reactivities of the functional groups in Molecule 1, Figure 4.13, are shown below:

```
NAME Amide
MEMBER 8
MEMBER 9
MEMBER 10
ACTIVE_BONDS 1
PERCENT 15.09
$$$$
NAME Amide
MEMBER 10
MEMBER 11
MEMBER 12
ACTIVE_BONDS 1
PERCENT 95.8567
$$$$
NAME Benzen
MEMBER 1
MEMBER 2
MEMBER 6
MEMBER 4
MEMBER 3
MEMBER 5
ACTIVE_BONDS 6
PERCENT 100
$$$$
NAME Ester
MEMBER 9
MEMBER 8
MEMBER 7
ACTIVE_BONDS 1
PERCENT 85.9
$$$$
NAME Amine
MEMBER 10
ACTIVE_BONDS 0
PERCENT 0
$$$$
NAME Ether
MEMBER 7
ACTIVE_BONDS 0
PERCENT 0
$$$$
NAME Alkene
MEMBER 1
MEMBER 2
MEMBER 3
MEMBER 4
MEMBER 5
MEMBER 6
ACTIVE_BONDS 3
PERCENT 0
$$$$
END
```

Figure 4.13: Result from the scoring program for Molecule 1

One challenge encountered during the designing of the scoring functions is an ability of the program to differentiate between a true symmetric functional group and a pseudo-symmetric functional group. This is critical in the prediction of the chemical reactions. Just because two

adjacent functional groups are identical, it cannot be assumed that they will behave identically. Their reactivity depends on the environment surrounding each of the functional groups. In Molecule 1, Figure 4.11, one of the amides consists of atom numbers 10, 11, and 12; while, the other consists of atom numbers 8, 9, 10. The first amide is expected to have a higher percent reactivity as amide than the second one. This is because the second amide shared its members with the near by ester (atom numbers 7, 8, and 9). Based on the result, it shows that the program is capable of verifying that the two amides are non-equivalent. The program assigns 96% reactivity to the terminal amide and only 15% to the internal amide. The 15% reactivity of the internal amide also indicates that the program acknowledges the presence of the ester, which shares more than half of its atoms with the amide. Another test molecule (Molecule 2, Figure 4.14) contains a symmetric amide, and is tested with the programs.

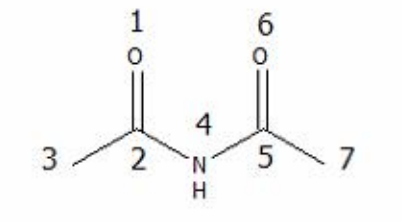


Figure 4.14: The structure of a test molecule (Molecule 2) that contains a symmetric functional group

The result is shown below:

```
NAME Amide
MEMBER 1
MEMBER 2
MEMBER 4
ACTIVE_BONDS 1
PERCENT 100
$$$$
NAME Amide
MEMBER 4
MEMBER 5
MEMBER 6
ACTIVE_BONDS 1
PERCENT 100
$$$$
END
```

Figure 4.15: Reactivity result for Molecule 2

Based on the results, it is clearly indicated that the strategy used in designing the scoring function is capable of differentiating the two forms of the functional group.

Based on the results in Figure 4.13, the 86% reactivity assigned to the ester functional group is higher than expected. This is due to the fact that the algorithm completely ignores the functional groups that have the same or a lower reactivity value than the target functional group. In this case, the ester has a reactivity value of 4.14 and the adjacent amide has a reactivity value of 4.01. If the algorithm was designed to take a difference between the two values, instead of completely ignoring the lower value, the ester would still have been selected as a more significant functional group than the amide, but with a lower percent reactivity. However, since the goal of the scoring function is to predict the functional groups that are likely to be active, the established procedure is considered satisfactory.

The functional group scoring program contains a function that determines the maximum number of shared atoms in the TFG. This is to ensure that the reactivity of the TFGs that are a part of larger functional groups, as in the case of the amine in Molecule 1, would automatically be assigned as zero, which implies that these TFGs are inactive. The zero reactivity of the alkene

functional group, Figure 4.10, whose members are atoms 1-6, indicates that the program recognizes the alkene atoms as parts of the benzene ring. The 100% reactivity assigned to the benzene functional group is a default value. This is because the chemistry of benzene depends on the types of substituients attached to the ring, as well as their relative positions to each other on the ring. Different substituients at different positions are likely to follow different reaction pathways. Therefore, the default value is assigned automatically to the functional group so that all of its reactions will be analyzed.

4.3 Conclusion

In this work, a set of functional group screening and scoring programs to be used with the chemical reaction prediction programs have been designed. These programs are capable of identifying the highly probable active functional groups in the given molecules. This allows the subsequent reaction prediction to be carried out only at the active functional groups. This eliminates the need of spending the computational time on predicting the reaction pathways of the non-active functional groups. Based on the obtained results, the programs are capable of differentiating between the true and pseudo-symmetric functional groups. The programs are also capable of assigning a satisfactory reactivity value to the ambiguous functional groups. One drawback of the programs is that they currently work well only with pre-defined functional groups. In reality, not all molecules contain only adjacent functional groups. Most molecules contain combinations of the known functional groups, as well as atoms whose connectivity patterns do not follow any of the functional groups. Therefore, as a part of our current progress, another set of algorithms are being designed to solve this problem.

REFERENCES

1. http://www.chemheritage.org/EducationalServices/pharm/chemo/activity/ester/ester01.gif
2. Mackie, R.; Smith, D. M.; Aitken, R., *Guidebook to Organic Synthesis (3rd Edition)*. Prentice Hall: 2000.
3. Wade, L. G., Jr., *Organic Chemistry (4th Edition)*. Prentice Hall: New Jersey, 1999.
4. http://homepage.smc.edu/kline_peggy/Organic/images/Organic_Compounds_ID.gif

Chapter 5: Term-Rewriting Grammar for Predicting Basic Chemical Reactions

Maude is a term-rewriting program that has been designed by the Formal Methods and Declarative Languages Laboratory at the University of Illinois at Urbana-Champaign.[1] The term transformation by Maude is carried out based on sets of pre-defined rules. Maude has two modules: system and functional. The system module performs term transformations based on rewriting rules; while, the functional module performs the transformations using algebraic equations. These modules can be used individually or collectively. Maude has been successfully used to model signaling pathway networks in mammals. [2] In this work, the system module is used. The term-rewriting strategy is used to directly assign reactions to the functional group patterns found in the small molecules. A set of separate auxiliary programs have also been designed, using a procedural programming language, to handle the changes in the connectivity information in the molecules as results of the reactions. This task separation allows the reaction prediction and the product construction to be carried out efficiently. The term-rewriting grammar is fast enough to be incorporated as a front-end program in a chemical database to supply chemical logic to a database program that otherwise can only handle strings and numbers. In this chapter, the designs of the term-rewriting rules for the prediction of simple organic reactions are described.

5.1 Chemical Term and Reaction Designs

5.1.1 Maude Programming for Chemical Reactions

The system module is declared using the starting and ending keywords *mod* and *endm*, respectively. In Maude, the programmer is allowed to define his/her datatypes and operators. This process must be designed carefully because it determines how the input terms are to be treated when they are encountered by the program.

The keywords for the datatype and operator declarations are sort(s) and op(s), respectively. The terms that are declared as constants are to be defined without underscores. For an example, the following term declaration is for a ***Process*** constant:

ops dehydration combination : -> Process .

Process is a sort that has been declared, and the terms ***dehydration*** and ***combination*** are two classified terms under ***Process***. The operators that are to carry out actions (term conversion, modification, etc.) are declared with underscores, '_'. For instance, consider the following declaration for the '+' operator:

op _+_ : ChemFeature ChemFeature -> ChemFeature .

ChemFeature (stands for 'chemical features') is a sort that has been declared in the module. The '+' operator is responsible for combining two ***ChemFeature*** terms to form a new ***ChemFeature*** term. The number of the datatypes used in the declaration must match the number of the underscores used. In the above example, there are two underscores for the '+' operator; therefore, two ***ChemFeature*** datatypes are required before the symbol '->'.

Different datatypes can also be combined:

op |__:_| : Process ChemFeature ChemFeature -> ChemFeature .

The |__:_| operator produced a ***ChemFeature*** datatype from two different datatypes (one ***Process*** and two ***ChemFeature***).

A complete system module that describes the dehydration and combination reactions of alcohol is shown below:

```
(1)   mod ACID is
(2)   sorts ChemFeature Process .
(3)   op _+_ : ChemFeature ChemFeature -> ChemFeature .
(4)   op |_:_| : Process ChemFeature ChemFeature -> ChemFeature .
(5)   op _=_ : ChemFeature ChemFeature -> ChemFeature .
(6)   op [_] : ChemFeature -> ChemFeature .
(7)   op {_} : ChemFeature -> ChemFeature .
(8)   ops dehydration combination : -> Process .
(9)   ops system acid alcohol ester amine amide : -> ChemFeature .
(10)  *** Rule Rewriting Section
(11)  rl { system = acid + alcohol } => | dehydration system : ester | .
(12)  rl { system = acid + amine } => | combination  system : amide | .
(13)  endm
```

Figure 5.1: The programming in the acid.maude

Lines 1 and 13 are the header and ending of the ACID module. Line 2 is the declaration of the two sorts (***ChemFeature*** and ***Processs***) that are used in this module. Lines 3-7 are the declaration of the operators, and lines 8-9 are the declaration of the ***Process*** and ***ChemFeature*** constant terms. The three asterisks in line 10 indicate that this line is a remark and is ignored by the program. The remark line can also be indicated by using three consecutive minus signs (---). Finally, lines 11 and 12 are the rules (indicated by the keyword "***rl***") that governs how the input terms (the terms on the left hand side of the implication symbol, =>) are to be treated. Note that all the terms and characters in the rules must be declared in the declaration section. Figure 5.2 below shows an example output from Maude:

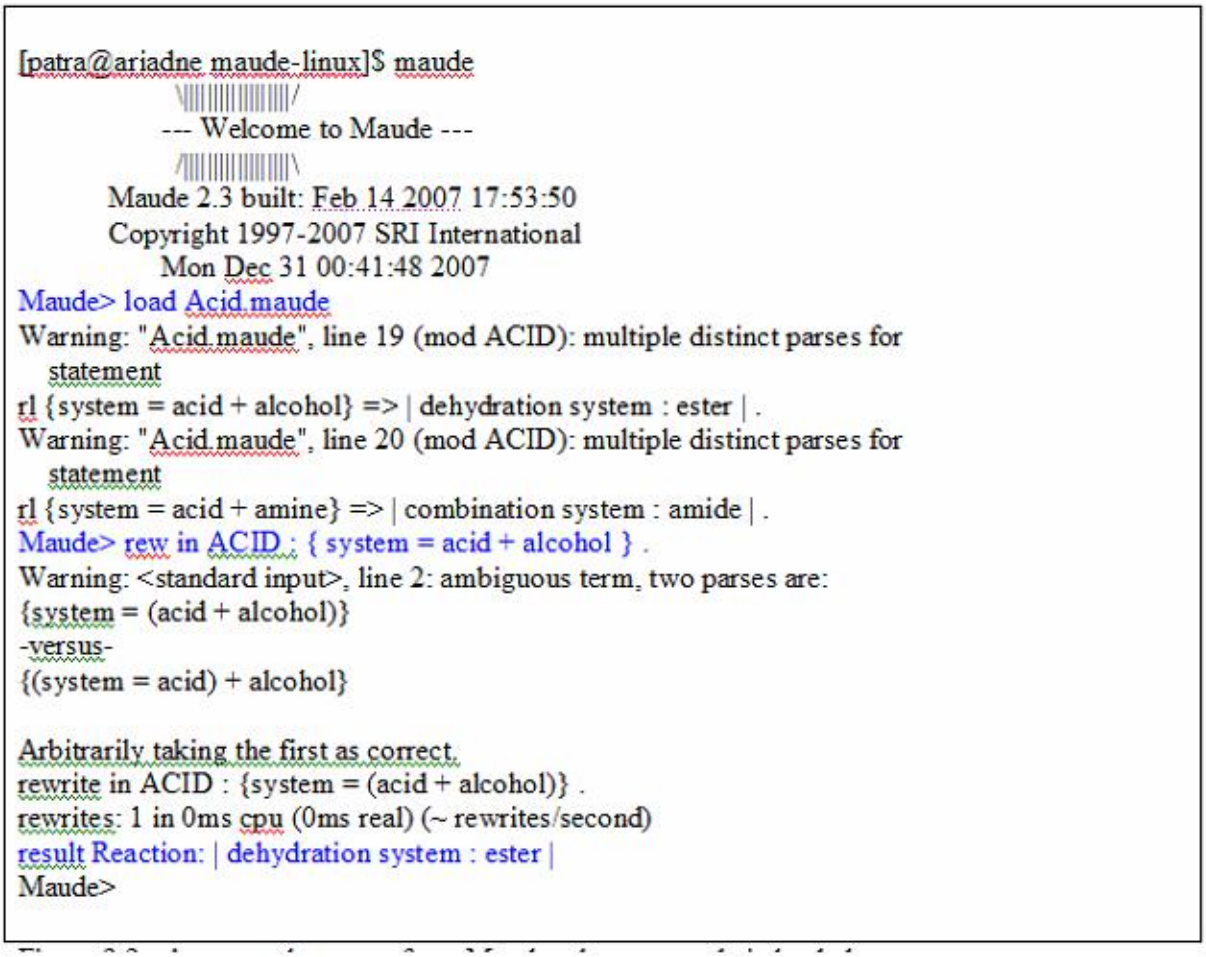


Figure 5.2: An example output from Maude when a .maude is loaded

The command line "load Acid.maude" uploads the Acid.maude file into the program. In this work, a .maude file contains reactions of a functional group. In the above example, Acid.maude contains reactions of the acid functional group. The command line "rew in ACID : { system = acid + alcohol } ." indicates the input { system = acid + alcohol } is to be analyzed by the rules in the ACID module. Finally, the line "result Reaction: | dehydration system : ester |" indicates the result because it returns the product description from Rule 11 in the ACID module. If no rules in the module applied to the input, a display of the original input will appear.

5.1.2 Designing of Terms

In this work, the reactions are classified into two groups: basic and advanced. The basic reactions refer to the functional group-to-functional group interactions. The reactions require only the presences of particular functional groups. It is, therefore, sufficient, in these cases, to use only the terms that represent the names of the functional groups. The following reaction, for example, is a nucleophilic reaction of a primary amine compound and an acid compound to form an amide, Figure 5.3.

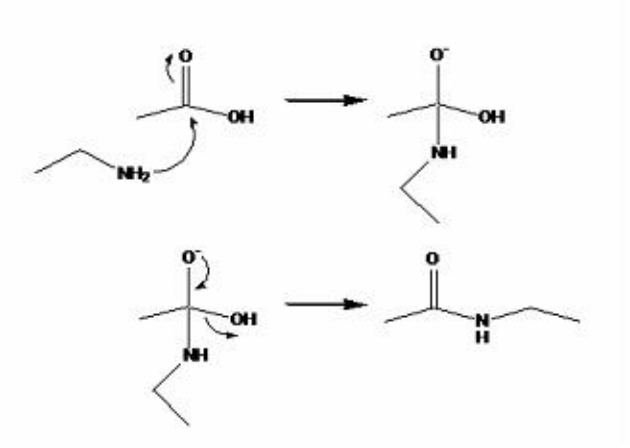


Figure 5.3: A nucleophilic reaction mechanism between an amine and a carbonyl compound

Since the reaction involves only the activities at the amine and the acid, the terms 'amine' and 'acid' are sufficient to represent the two reactants in this system. The rest of the compounds can be omitted from the representation because they are not involved in the reaction.

The advanced reactions require more information than just the presences of the functional groups. Consider the Baeyer-Drewson Indigo synthesis that is shown below.

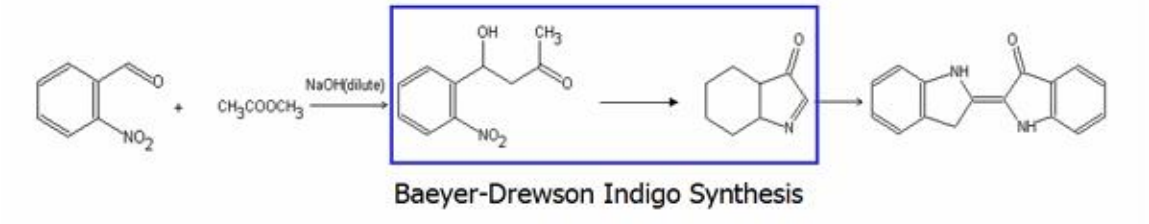


Figure 5.4: Baeyer-Drewson Indigo Synthesis [3]

The synthesis reaction, shown in Figure 5.4, requires the nitro (-NO_2) to be positioned in such a way that it can interact with the methyl (-CH_3) to form the five-member ring. Advanced reactions may not always require the same information. Some may require a group of atoms to be arranged in a specific pattern; while, others may require only a presence of a specific atom with a particular atomic geometry. To handle such a variety of required information, the terms for the advanced reactions are designed such that they collectively represent the critical features, or the reactive portion in the molecule, for the reactions. This task is accomplished by using the atoms.sp4 atomtypes to describe the atoms that are involved in the reactions.

An advantage of using the atomtypes is that the atomic geometry is included. This allows the backbone atoms, or the atoms that are most likely to be involved in the reactions, to be labeled.

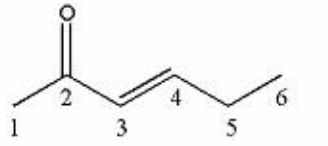


Figure 5.5: An example molecule that shows how the atomtypes can be used to label the atoms in the molecule

One possible set of terms that can be used to describe the test molecule in Figure 5.5 is (C3-1, C2-2, C2-3, C2-4, C3-5, C3-6). C3-1 represents the sp3 carbon at the position 1, C2-2 represents the sp2 carbon at the position 2, and so forth. However, confusion may arise when two different types of atoms share the same geometry. In the above example, C2-2 and C2-3 have identical atomic geometry, even though C2-2 is a carbonyl carbon and C2-3 is an alkene carbon. To differentiate the two, additional descriptive terms can be added. This produces a new term set for the molecule: (C3-1, carbonyl-C2-2, alkene-C2-3, alkene-C2-4, C3-5, C3-6). The

use of the atomtypes to describe the atoms is optional, depending on the requirement of the reactions. Other terms that are more chemically specific can be used if they can describe the reaction requirements more accurately.

5.1.3 Designing of Reaction Rules

The general expression of a chemical reaction is:

$$A + B \Rightarrow C$$

The terms A and B are *reactants*, or starting materials, of the reaction, and the term C is a *product*, or a result, of the reaction. The term-rewriting rule is designed to mimic the expression to retain the representations as closely as possible to those in the literature.

For the basic reactions, it is sufficient to declare the rule using the names of the interacting functional groups. For instance, a dehydration reaction between an acid and an alcohol functional groups yield an ester. This rule can be expressed as:

rl { system = acid + alcohol } => | dehydration system : ester | .

The keyword *rl* informs the Maude program that the following syntaxes are a term transformation rule. The keyword *system* indicates that the target system contains *acid* and *alcohol*, and the system components are enclosed in *{}*. |__:_| is an operator that describes the reaction. In this rule, | *dehydration system : ester* | indicates that acid and alcohol can form an ester through dehydration reaction. The keyword *system* in the product serves the same purpose as in the reactant. The general format for the rule expression for the functional group-to-functional group reactions is:

rl { system = reactant descriptions } => | process_name system : product descriptions | .

Process_name refers to the name of the reaction or process that converts the reactant to the specified product. For the reactions that have more than one step, the number of rules developed corresponds to the number of the required steps. If there are four steps, for example,

required for the conversion of a reactant to a product, four separate rules are developed. The *process_name* is associated to a specific set of product modeling functions. This ensures that the product from each step is accurately modeled. The product descriptors are not involved in the structure modeling. They convey significances in the intermediate structures. Once generated, their representation is processed by an auxiliary program and becomes an input or a part of an input for the next applicable reaction. This continues until the last applicable reaction rule is reached.

For a multiple-step reaction, the rules are divided into separate self-contained modules. The Maude language permanently replaces the old term with the new term as each rule is applied. Hence, if there are several rules that are applicable to an input, only the first rule is executed. Once the input representation has been modified, it is no longer recognizable by the subsequent rules. The module separation technique ensures that all of the reactions that recognize the same reactant will be executed.

The reaction expression for the advanced reactions is similar to that of the basic reactions. The advanced reactions can be expressed as follows:

$$(R_1 + R_2 + R_3 + \ldots + R_N) \Rightarrow |\, process\_name : P_1 + P_2 + P_3 + \ldots + P_M \,|\,.$$

where R = Reactants, N = total number of the reactants, P = Products, and M = total number of the products

The reactants and the products on different compounds are separated by '+' operator. Each of the reactant R can be any critical feature that is crucial to the reaction. Likewise, each of the product P can be any of the transformed features that resulted from the reaction. For instance,

each of the terms in (C3-1, C2-2, C2-3, C2-4, C3-5, C3-6) that describe the molecule in Figure 5.5 is a recant R. Each of the reactant can be expanded as follows:

$(T_1\text{-}R_1, T_2\text{-}R_1, \ldots, T_A\text{-}R_1) + (T_1\text{-}R_2, T_2\text{-}R_2, \ldots, T_A\text{-}R_2) + \ldots + (T_1\text{-}R_N, T_2\text{-}R_N, \ldots, T_A\text{-}R_N)$

where R = reactant, T = term, N = reactant number, A = term number

For instance, in the term carbonyl-C2-2 in (C3-1, carbonyl-C2-2, alkene-C2-3, alkene-C2-4, C3-5, C3-6), 'carbonyl' is a descriptive term for the reactant C2-2. The same technique can also apply to better describe the products.

5.2 Structure Handling Programs

Maude works only with string inputs. Although the program has been used for a number of event-driven predictions, it is difficult to use Maude to model systems that involve connectivity specification, like the chemical reactions. An ability to modify the molecule at the active location, while maintaining the connectivity of the rest of the molecule, is important in this work. Hence, a set of structure handling programs was created to work in conjunction with Maude to accurately construct the predicted product. This structure handling section consists of two parts: functional group pattern generation and product structure modeling. The programs in both parts are written using C++ programming language (Microsoft Visual C++ 6.0).

The test structures are created using ChemDraw Ultra 6.0 program in the ChemOffice 6.0 package. The structures are saved as a MDL .mol files. The functional groups in each of the test files are screened and scored using the programs discussed in Chapter 2. The results obtained from the scoring program are used by the *functional group pattern generation programs* (FGPGPs) to generate the proper Maude inputs. The results from the executed Maude programs

are then sent to the modeling unit that consists of the *product modeling programs* (PMPs), which automatically build the desired products.

5.2.1 Functional Group Pattern Generation Programs (FGPGPs)

A FGPGP is created for each of the functional groups. Only the FGPGPs whose functional group is predicted to have at least 30% reactivity are executed. The percent reactivity information is obtained from the percent profile generated from the scoring program described in Chapter 2. The minimum reactivity requirement can be adjusted as appropriate. In addition to the percent profile, the original .mol structure file is also input into the selected FGPGPs to keep the connectivity information in the molecule consistent. This allows the FGPGPs to operate immediately at their target atoms without spending additional time searching for them.

The FGPGPs have a full knowledge of the patterns required by the Maude programs. Each of the FGPGPs examines their target atoms to determine if their connectivity matches any of the required patterns. If there is a match, the pattern is generated. For instance, Alcohol.maude contains three modules: oxidation, reduction, and dehydration. The rules declared in the modules are shown in Table 5.1:

Table 5.1: The three modules in the Alcohol.maude program and their rules

Module Name: Oxidation rl { system = alcohol } => | oxidation system : ketone | .
Module Name: Reduction rl { system = alcohol } => | reduction system : alkane | .

Module Name: Dehydration

rl { system = { hydroxyl-carbon = alcohol-O + alkane-neighbor } - { alkane-neighbor = hydroxyl-carbon + H + R + R } } => | dehydration system : alkene | .

If a given molecule is known to contain an alcohol functional group, whose reactivity is at least 30%, then FGPGP for alcohol is executed to search for two input patterns: *alcohol* (for the Oxidation and Reduction modules) and *{ hydroxyl-carbon = alcohol-O + alkane-neighbor } - { alkane-neighbor = hydroxyl-carbon + H + R + R }* (for the Dehydration module). The first pattern is straightforward and corresponds to an hydroxyl oxygen. However, for the Dehydration module, two requirements must be satisfied: (1) the hydroxyl carbon (*hydroxyl-carbon*) must be present, and (2) one of its neighbors (*alkane-neighbor*) must carry at least one hydrogen atom. *{ alkane-neighbor = hydroxyl-carbon + H + R + R }* signifies that as long as this *alkane-neighbor* attaches to at least one hydrogen atom ("*H*"), the remaining of the non-specified neighbors can be ignored and are assigned a default identity "*R*". If both of the requirements are satisfied, two separate output files are generated. One is a .txt file containing the atomic numbers (as appeared in the original .mol file) of the involved atoms. Another output contains the Maude input *{ system = { hydroxyl-carbon = alcohol-O + alkane-neighbor } - { alkane-neighbor = hydroxyl-carbon + H + R + R } }*. Notice that the terms do not use the atomtypes. This is because the terms used here provide better descriptions to the reaction.

5.2.2 Product Modeling Programs (PMPs)

A set of product modeling programs is created for each reaction. For the three alcohol reactions, three sets of the product modeling programs are created. There are three required

inputs for the product modeling: (1) the original .mol file of the molecule, (2) the atomic number information generated by the FGPGPs, and (3) the output from the Maude program.

The first file provides original connectivity information of the molecule. The second file informs the modeling programs the exact locations of the atoms to be modified in the molecule. Finally, the reaction name indicated in the Maude output files determines which of the modeling sets is to be executed. The modeling process does not require the product descriptors. This is because the modification instructions are too complicated to be encoded into a line of string. Condensing the connectivity information, for example, into a string and reconverting it back such that it can be used in the product construction can lead to some lost of the original information. It is, therefore, more practical to design an independent modeling unit that can immediately construct the molecules without the conversions.

For instance, if the PMPs for the alcohol dehydration is executed, the following procedures take place:

(1) the hydroxyl oxygen is disconnected from the hydroxyl-carbon

(2) the connectivity in the input molecule is updated

(3) a double bond is formed between the hydroxyl-carbon and the neighbor-alkane

(4) the connectivity in the input molecule is updated

(5) a new .mol file for the newly formed product is generated

The steps taken in the PMP are not necessarily chemistry-based. The goal here is to quickly build the desired product. This is a valid approach because each of the rules focuses on a step-by-step modification.

5.2.3 Modeling of Products of Two Compounds

The PMPs that construct products of two separate reactants follow the same procedures as with one reactant. The two example reactions are shown below.

rl { system = acid + alcohol } => | dehydration system : ester | .

rl { system = acid + amine } => | combination system : amide | .

Thus far, when two reactants are predicted, it is assumed that two compounds are involved. In these cases, required input files are obtained separately for each of the .mol files. However, the FGPGPs clearly indicate, in the atom number .txt file, the functional group roles of the compounds. The PMPs created for the reactions are designed to recognize this information. They know which of the compounds behaves as an acid, for example, an alcohol, or an amine. The prediction of the products then proceeds as previously described.

5.3 Results

The focus of this part of the work is to develop a term-rewriting rule expression that is capable of representing a wide range of reactions. Several reactions have been used to test the design. They can be divided into two categories: the basic functional group reactions and the advanced chemical reactions.

5.3.1 Basic Functional Group Reactions (BFGRs)

The basic functional group reactions are the commonly known reactions that occur at or between the functional groups. These include the self-modification reactions (such as oxidations, reductions, and dehydrations), as well as those reactions that occur between functional groups. Some of the example functional group rules are shown below:

rl { system = ester } => | hydrolysis system : acid | .

rl { system = acid + alcohol } => | dehydration system : ester | .

rl { system = acid + amine } => | combination system : amide | .

rl { system = ester } => | reduction system : alcohol | .

rl { system = ester + amine } => | combination system : amide | .

rl { system = ester + alcohol } => | combination system : ester | .

rl { system = amide } => | hydrolysis system : acid | .

rl { system = amide } => | reduction system : amine | .

The rules are organized into individual .maude files according to their functional groups, and further subdivided, in each of the files, into individual modules. Ester has been chosen as a test functional group because it can go through several reactions to generate different products that can proceed through further reaction pathways. The goal here is to analyze the performance of predicting reaction pathways of a given starting functional group. The ester functional group is represented by a term *ester*, and its Maude input representation is *{ system = ester }*. The input is entered into the three modules in the Ester.maude (Appendix C)

The input is entered into the modules individually, as shown below:

```
Maude> rew in ESTER1 : { system = ester } .
rewrite in ESTER1 : {system = ester} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: | hydrolysis system : acid |

Maude> rew in ESTER2 : { system = ester } .
rewrite in ESTER2 : {system = ester} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: | reduction system : alcohol |

Maude> rew in ESTER3 : { system = ester } .
rewrite in ESTER3 : {system = ester} .
rewrites: 0 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: {system = ester}
```

The blueprints are the results from the modules. Both modules ESTER1 and ESTER2 return the results: | *hydrolysis system : acid* | and | *reduction system : alcohol* |. The results are

indicated by the bar symbol "|", followed by the reaction name, a system specification term, and the name of the product functional group(s). The terms *acid* and *alcohol* represent the resulting functional groups in the products. Since none of the rules in module ESTER3 recognizes the input, the original pattern is returned. This output is ignored and will not be further processed. The interpretation of the results is that, within this defined scope of the reactions, the ester functional group can go through two reaction pathways: the hydrolysis to form acid, or the reduction to form an alcohol. Since acid and alcohol also have their own sets of reactions, the output terms are re-organized into the Maude input format and allowed to continue with their reaction pathways.

The acid reactions defined in the Acid.maude shown in Appendix D. The following show the results obtained from the modules when the new acid input *{ system = acid }* was entered:

```
Maude> rew in ACID1 : { system = acid } .
rewrite in ACID1 : {system = acid} .
rewrites: 0 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: {system = acid}
Maude> rew in ACID2 : { system = acid } .
rewrite in ACID2 : {system = acid} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result Reaction: | reduction system : aldehyde |
Maude> rew in ACID3 : { system = acid } .
rewrite in ACID3 : {system = acid} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result Reaction: | reduction system : primary-alcohol |
Maude> rew in ACID4 : { system = acid } .
rewrite in ACID4 : {system = acid} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result Reaction: | alkylation system : ketone |
```

As expected, only modules ACID2, ACID3, and ACID4 return the results. Module ACID1 does not contain *{ system = acid }* terms, and, therefore, its rule does not apply to the input. The same procedures are applied to the alcohol input with the rules in Alcohol.maude. The rules in Alcohol.maude are organized in a similar manner to those in the Acid.maude.

To examine the program's behaviors when two functional groups are used, an alcohol is added into the acid system. The input is now *{ system = acid + alcohol }*. When this input is re-entered into the acid modules, only the ACID1 module returns a result:

```
Maude> rew in ACID1 : { system = acid + alcohol } .
rewrite in ACID1 : {system = (acid + alcohol)} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result Reaction: |dehydration system : ester |
Maude> rew in ACID2 : { system = acid + alcohol } .
rewrite in ACID2 : {system = (acid + alcohol)} .
rewrites: 0 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: {system = (acid + alcohol)}
Maude> rew in ACID3 : { system = acid + alcohol } .
rewrite in ACID3 : {system = (acid + alcohol)} .
rewrites: 0 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: {system = (acid + alcohol)}
Maude> rew in ACID4 : { system = acid + alcohol } .
rewrite in ACID4 : {system = (acid + alcohol)} .
rewrites: 0 in 0ms cpu (0ms real) (~ rewrites/second)
result ChemFeature: {system = (acid + alcohol)}
```

Similarly, if the term *alcohol* is replaced with a new term *amine*, which makes the input *{ system = acid + amine }*, only the first module responds to the input, and the result is:

```
Maude> rew in ACID1 : { system = acid + amine } .
rewrite in ACID1 : {system = (acid + amine)} .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result Reaction: |combination system : amide |
```

Each of the test inputs is applied to all programs and their modules. The approach bypasses the need of additional programs to determine which of the Maude programs are to be activated. Only the rules that recognize the input patterns are expected to respond. The outcomes can, then, be combined to construct reaction pathways of the starting functional group. The results from the two-step experimental simulation suggest that this strategy has satisfied the objective. Converting the product descriptors to the reactant descriptors allows the pathway to continue from one set of functional group reactions to another. However, when more than one product is formed, the product terms must be organized into the correct order before they can be

further applied to other rules. In other words, the representations "*acid + alcohol*" and "*alcohol + acid*" must be determined as being equivalent. This can be established via designing an additional program that determines the equivalency or additional rules can be added in the modules such that more than one representation can be generated. An overall reaction pathway of the ester is graphically summarized as shown in Figure 5.6:

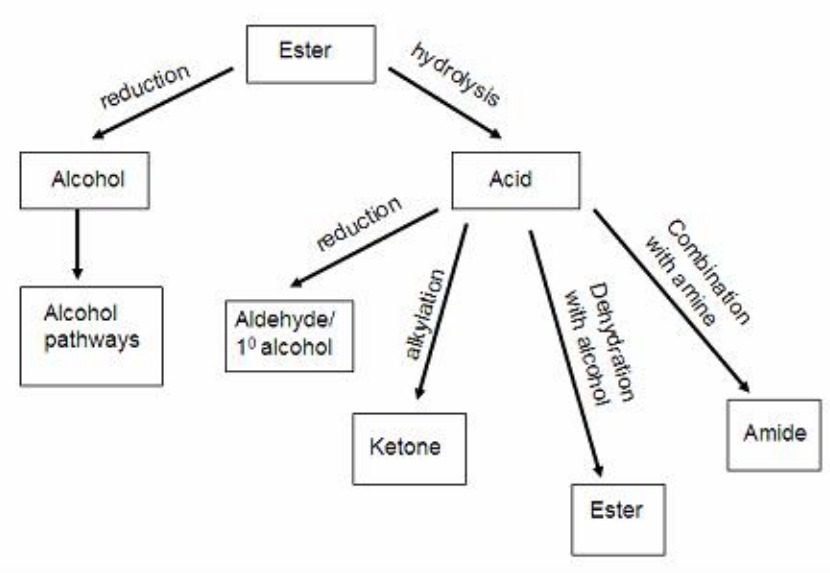


Figure 5.6: An overall reaction pathways of an ester

5.3.2 Advanced Chemical Reactions

A. Mechanism-Emphasis Reactions: Substitutions and Eliminations Mechanisms

The substitution (SN1 and SN2) and the elimination (E1 and E2) mechanisms are four of the fundamental of mechanisms in organic chemistry. The factor that determines the reactions that a compound will go through is the nature of the reactive carbon. Sn2 reactions require that the reactive carbon is either a primary or a secondary carbon, Figure 5.7.

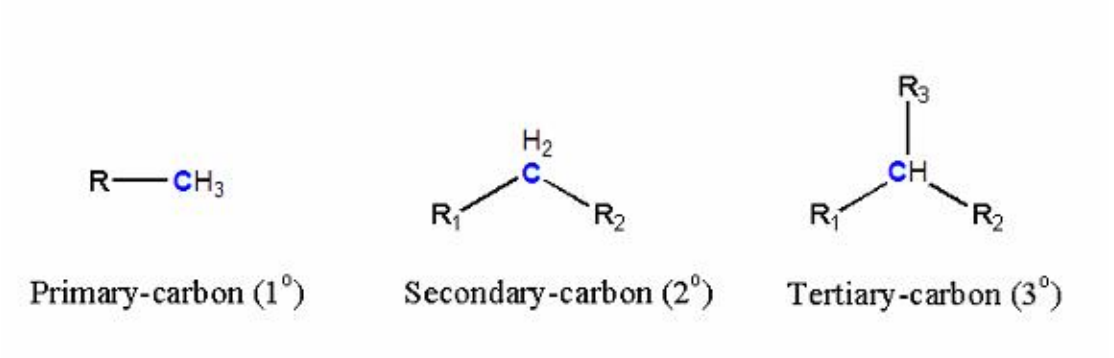


Figure 5.7: The descriptions of the primary, secondary, and tertiary carbons

These carbons have less steric hindrance than the tertiary carbon. Sn2 is a single-step reaction that shifts the stereo-configuration. Hence, the less the steric hindrance at the target carbon, the more favorable is the reaction, Figure 5.8.

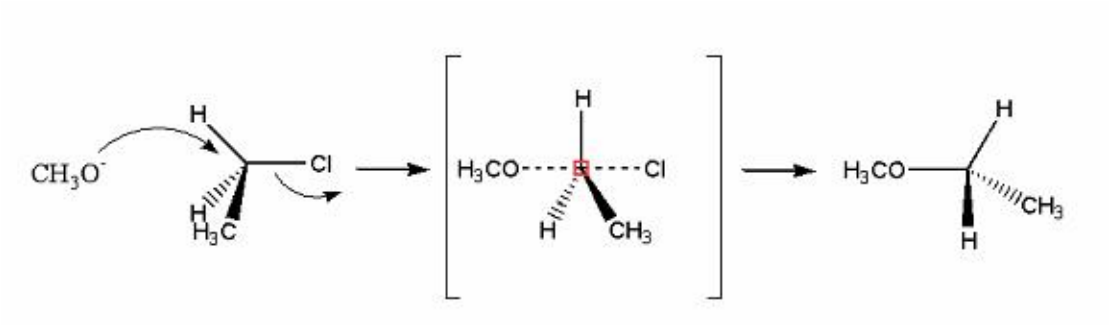


Figure 5.8: The back-side attack in the Sn2 reaction

Tertiary carbon is too great a steric-hindrance and the nucleophile cannot form the right orientation at the carbon to carry out the SN2 mechanism, Figure 5.9.

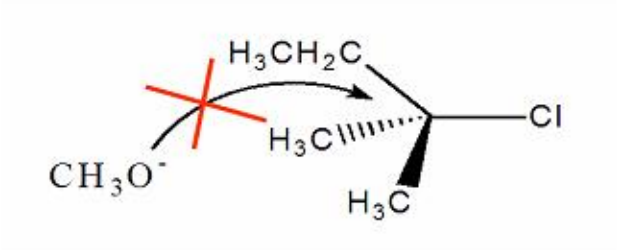


Figure 5.9: The steric hindrance of the tertiary carbon prevents the nucleophile from attacking the carbon

On the other hand, the SN1 reaction favors tertiary carbons. In SN1, the leaving group first leaves the compound to become an independent product. This provides the parental compound with an extra space around the carbon, which allows the nucleophile to attack, Figure 5.10.

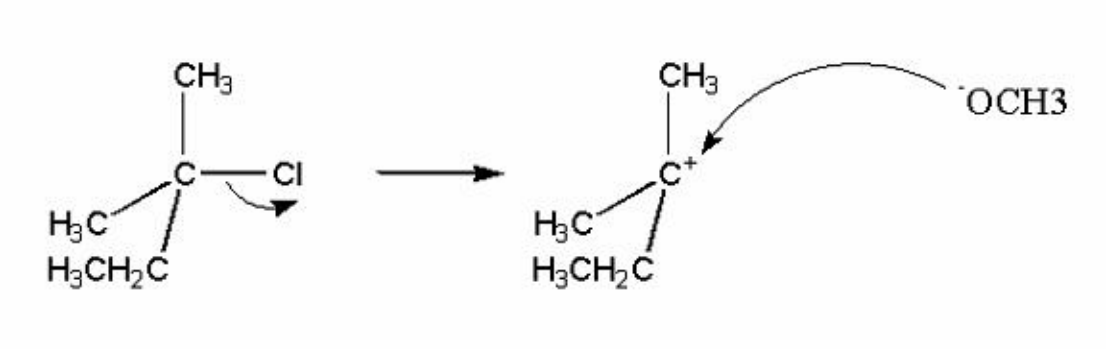


Figure 5.10: The mechanism of the nucleophilic reaction with the tertiary carbon

The elimination reactions share their pathways with the substitution reactions. The factor that determines if the elimination would take place is the neighbors of the reactive carbon. If one of the neighbors carries at least a hydrogen, then elimination can be expected.

E1 reaction occurs simultaneously with the SN1. This is because the nucleophile can either attack directly at the cation carbon, or strip off the hydrogen from the neighboring carbon, Figure 5.11.

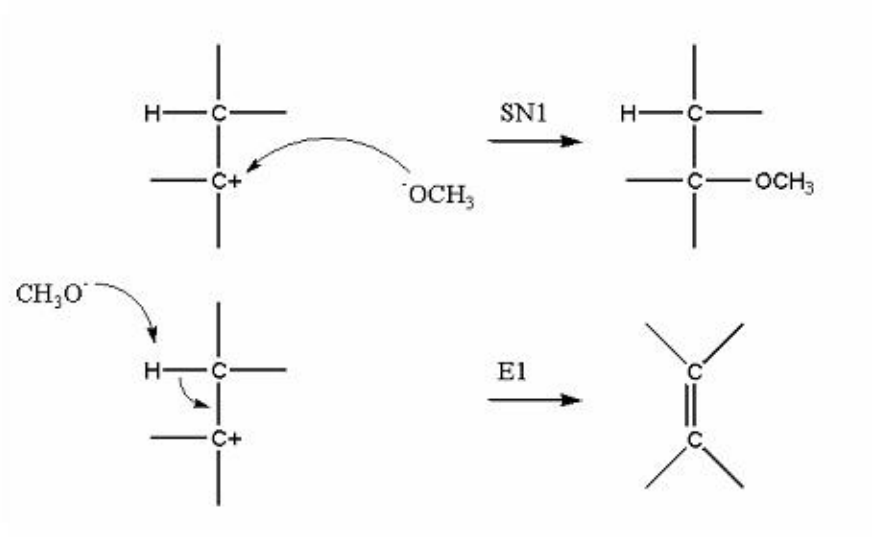


Figure 5.11: The pathways of the SN1 and E1 reactions

E2, on the other hand, can occur at the secondary- or the primary-carbon, Figure 5.12. E2 reactions usually take place with a strong nucleophile.

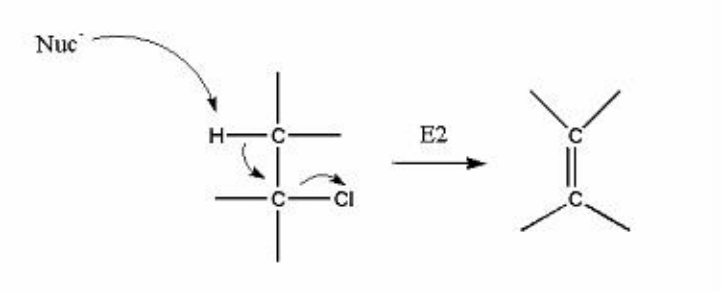


Figure 5.12: The E2 mechanism

The followings show the rules describing the substitution and elimination reactions:

SN2 and E2:

(1) rl { system = primary-carbon - carbon-bearing-H-neighbor } => | halide-substitution system : { carbon = carbon-bearing-H-neighbor + H + H + halide } | .

(2) rl { system = primary-carbon - R } => | halide-substitution system : { carbon = R + H + H + halide } | .

(3) rl { system = { carbon = carbon-bearing-H-neighbor + H + H + halide } + { nucleophile } } => | E2 system : alkene | .

(4) rl { system = { carbon = R + H + H + halide } + { nucleophile } => | S2 system : carbon = R + H + H + nucleophile | .

(5) rl { system = secondary-carbon - carbon-bearing-H-neighbor } => | halide-substitution system : { carbon = carbon-bearing-H-neighbor + R + H + halide } | .

(6) rl { system = secondary-carbon – R1 } => | halide-substitution system : { carbon = R1 + R2 + H + halide } | .

(7) rl { system = { carbon = carbon-bearing-H-neighbor + R + H + halide } + { nucleophile } } => | E2 system : alkene | .

(8) rl { system = { carbon = R1 + R2 + H + halide } + { nucleophile } => | S2 system : carbon = R1 + R2 + H + nucleophile | .

Rules 1-4 apply to the reactions at the primary-carbon, and rules 5-8 apply to the secondary carbon. Rules 1 and 2 differentiate between a primary carbon that has a hydrogen-bearing neighbor (Rule 1) and those that do not (Rule 2). The term '*R*' in Rule 1 is a place holder. It signifies that the portion connected to the carbon is not significant for the reaction. The minus sign indicates that the *R* term is directly connected to the *primary-carbon*. Both cases undergo a halide substitution. However, only those that have a hydrogen-bearing neighbor go through E2 reaction (Rule 3); while, those that do not have such a neighbor undergo SN2 reaction (Rule 4). Both the elimination and substitution require a nucleophile. The *nucleophile* is enclosed in a separate set of {} brackets because it is a separate compound. Rules 5-8 are a

repetition of rules 1-4 and apply to a secondary-carbon. The *R* terms in rules 5-8 serve the same function as place holders as in rules 1-4.

The following shows that rules for SN1 and E1 reactions:

SN1 and E1

(1) rl system = { tert-carbon – carbon-bearing-H-neighbor } => |cation-formation system : charged-tert-carbon - carbon-bearing-H-neighbor | .

(2) rl system = tert-carbon – R1 } => |cation-formation system : charged-tert-carbon – R1 | .

(3) rl system = { charged-ter-tcarbon – carbon-bearing-H-neighbor } + { nucleophile } => | E1 system = alkene | .

(4) rl system = { charged-tert-carbon – carbon-bearing-H-neighbor } + { nucleophile } => | SN1 system = tert-carbon = nucleophile + carbon-bearing-H-neighbor + R1 + R2 | .

(5) rl system = { charged-tert-carbon – R1 } + {nucleophile } => | SN1 system : tert-carbon = nucleophile + R1 + R2 + R3 | .

These rules are designed for tertiary carbon, and their design is similar to those for SN2 and E2. Rules 1 and 2 describe the cation-formation reaction for the tertiary carbon that has a neighbor with a hydrogen atom (Rule 1) and that does not have a hydrogen-bearing neighbor (Rule 2). The term *charged-tert-carbon* represents the resulting charged tertiary carbon. Rules 3 and 4 both apply to the carbon with a hydrogen-bearing neighbor. This is because E1 and SN1 occur simultaneously with one another. However, for the carbon without a hydrogen-bearing neighbor, only SN1 reaction is expected.

Rules 1, 2, 5, and 6 are included into the first module. The rules can be placed within the same module because they recognize different initial input patterns. Rules 3, 4, 7, and 8 are organized into different modules because they deal with intermediate patterns. The module separation is beneficial particularly to rules 4 and 8, where the term *nucleophile* is used. Not all

compounds play the same role in the reactions. Some may behave as a nucleophile for a group of reactants, while remaining non-reactive toward the others. Hence, dividing reactions into different modules allow each of the reactions to contain their own set of nucleophiles. The test structures for the SN2 and E2 reactions are shown below, Figure 5.13:

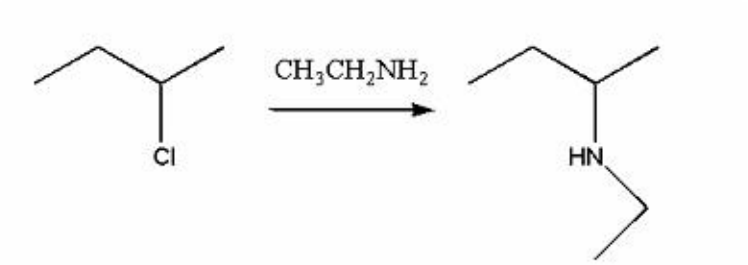


Figure 5.13: Nucleophilic reaction between the amine and the 2-chloro-butane

The above reaction is expected to go through SN2 and E2 processes, and the rules describing the rules are as followed:

(1) rl (secondary-atom , chlorine , hydrogen-bearing-neighbor) =>

(secondary-atom, leaving-group , hydrogen-bearing-neighbor) .

(2) rl (propylamine) => (nucleophile) .

(3) rl (secondary-atom, leaving-group , hydrogen-bearing-neighbor) + (nucleophile) =>

[SN2 + E2] .

The rule in line 1 indicates an internal conversion of the *chlorine* to a *leaving-group*, and the rule in line 2, similarly, indicates an conversion of the *propylamine* to a *nucleophile*. The following shows the returned result when the input *(secondary-atom , chlorine , hydrogen-bearing-neighbor)* is entered into the module SCREENING:

Maude> rew in SCREENING : (secondary-atom , chlorine , hydrogen-bearing-neighbor) + (propylamine) .
rewrite in SCREENING : (secondary-atom,(chlorine,hydrogen-bearing-neighbor)) + propylamine .
rewrites: 3 in 0ms cpu (0ms real) (~ rewrites/second)

result [Reaction]: [SN2 + E2]

The obtained result indicates that the terms *chlorine* and *propylamine* have successfully been converted to *leaving-group* and *nucleophile*, resprectively, as expected. In this module, the conversion rules are placed before the reaction rules. The result also indicates that this method allows the overall reaction to be predicted.

The rule separation for the SN1 and E1 is similar to those in SN2 and E2. Rules 1 and 2 are grouped into the first module to recognize the input patterns. Rules 3, 4, and 5 are divided into their own modules. The test structure for SN1 and E1 is shown in Figure 5.14:

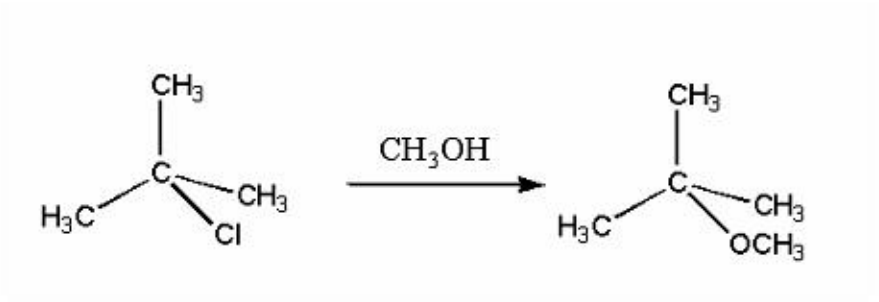


Figure 5.14: The substitution (SN1) reaction of the methanol and 2-chloro-tert-butane

The above reaction is expected to go through SN1 and E1 processes. The rules describing the two reactions are as shown below:

(1) rl (tertiary-atom , chlorine , hydrogen-bearing-neighbor) =>

(tertiary-atom , leaving-group , hydrogen-bearing-neighbor) .

(2) rl (methanol) => (nucleophile) .

(3) rl (tertiary-atom , leaving-group , hydrogen-bearing-neighbor) + (nucleophile) =>

[SN1 + E1] .

The above rules are also written in the SCREENING module. As in the previous example, the rule in line 1 indicates the conversion of *chlorine* to *leaving-group*, and the rule in line 2 indicates

the conversion of *methanol* to *nucleophile*. The result when *(tertiary-atom , chlorine , hydrogen-bearing-neighbor)) + (methanol)* entered is:

Maude> rew in SCREENING : (tertiary-atom , chlorine , hydrogen-bearing-neighbor) + (methanol) .
rewrite in SCREENING : (tertiary-atom,(chlorine,hydrogen-bearing-neighbor)) + methanol .
rewrites: 3 in 0ms cpu (0ms real) (~ rewrites/second)
result [Reaction]: [SN1 + E1]

The result indicates that the internal conversion of the terms chlorine and methanol have been successfully applied. The SN1 and E1 rules are placed after the SN2 and E2 processes. The result indicates that program immediately applied the rules for the SN1 and E1 processes.

The main reason that SN1 reaction is favorable at the tertiary carbon is because the cation is most stable at the tertiary position. The stability trend of a cation at the carbon can be generally expressed as followed:

$$3° > 2° > 1° > \text{methyl}$$

However, there is an exception to the rule. A cation can also be formed at a primary and a secondary carbon, if there is a more stable carbon nearby. This phenomenon is called a hydride-shifting. Consider the following structure in Figure 5.15. C2 is a secondary carbon that is attached to a chlorine and a tertiary-carbon (C3).

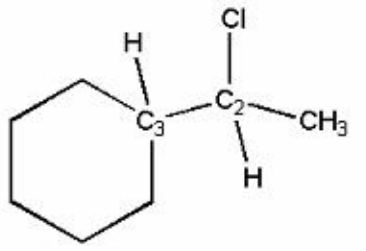


Figure 5.15: 2-chloro-ethylcyclohexane

Here, the cation formation at the tertiary C3 carbon would be more stable than at the secondary C2. However, the leaving group chlorine is attached at the C2. The chlorine is forced to leave

since the hydrogen can shift to the secondary carbon to form a hydride (:H) tertiary carbon, Figure 5.16. This places the cation on the tertiary carbon and allows the compound to go through the usual Sn1 and E1 mechanisms.

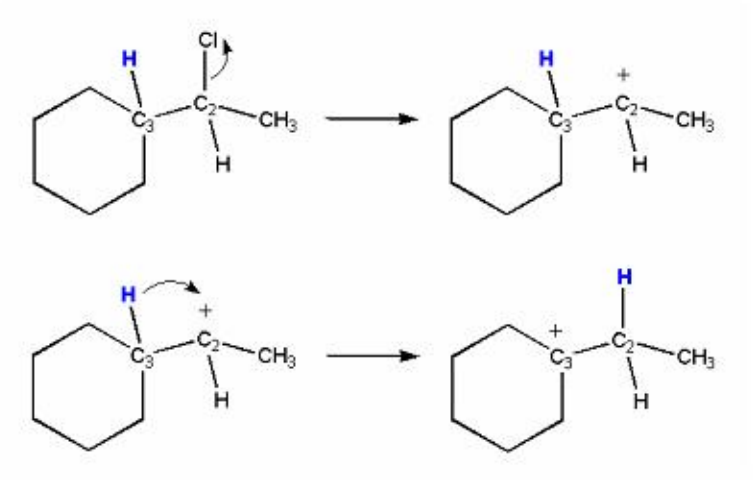


Figure 5.16: Hydride shifting between the clorine (Cl) and the hydride (blue H)

This process is called a 1,2-hydride shift. (It is called a 1,2-shift because the hydrogen atom moves from one atom to the adjacent atom)

The processes in the hydride shifting can be described using the rules below:

(1) rl { system : secondary-carbon-halide - tert-carbon-with-hydrogen } => | cation-formation system : charged-secondary-carbon – tert-carbon-with-hydrogen | .

(2) rl { system : charged- secondary-carbon - tert-carbon-with-hydrogen } => | hydride-shifting system : secondary-carbon – charged-tert-carbon | .

(3) rl { system : secondary-carbon – tert-charged-carbon } => | term-change system : charged-tert-carbon – carbon-bearing-H-neighbor | .

Rule 1 describes the initial state of the starting compound, and Rule 2 describes the hydride-shifting process. Additional Rule 3 is needed because once the cation has migrated to the tertiary carbon, it can enter the normal SN1 and E1 pathways. Hence, in Rule 3, the left-side

terms are changed to the right-side terms, which is the exact input pattern for Rule 1 in SN1 and E1 reactions.

5.4 Discussion

We have shown that the rewriting grammar, when designed properly, can be used to develop the rewriting rules that mimic the rules of chemical reactions. The grammatical approach uses the method of pattern recognition, instead of the condition specifications as required by the procedural programming. This allows the chemically significant features in the molecule, instead of the whole molecule, to be expressed. It also allows the term-rewriting programs to immediately assign the reactions that are applicable to the patterns. When used in conjunction with procedural programming, the predicted product structures can be quickly constructed. This is critical in the situations where a large number of input samples are handled continuously. In these cases, this new approach can greatly reduce the computational time, as well as maintain the prediction accuracy.

The major challenge encountered during the design is the information passing between the Maude and the procedural programs. The goals of this work are to predict the reactions that are applicable to the detected patterns, and to construct their products. The reaction assignment is performed by the Maude language, which recognizes only the string input format. On the other hand, the product construction unit is handled by procedural programming that requires the connectivity knowledge from the original input file, which is in a .mol structure format. It is difficult to design a program that converts data between the two formats without losing some of the original information. Furthermore, putting all of the structural information into the string format can be overwhelming, particularly with the structure-specific reactions. The solution to this problem is to allow all of the procedural programs to access the original input files. However, those programs for the product construction require the permission, or the matched

reaction names, generated by the Maude programs to proceed with the product construction. This strategy both eliminates the need for embedding all of the structural information in the string, and maintains the connectivity information by allowing an immediate access to the original input file to the programs, when needed.

Despite the advantages of this method, several issues must be considered when designing the terms and rules for complicated structure-specific reactions. In these reactions, atoms in various states (such as charged metals, anions, cations, intermediate carbon, etc) may be present. Choosing proper terms and representations that are chemically correct is important for result validation. Also, with larger reactants and intermediates, lengthy and complicated representations may be unavoidable, which may lead to confusion.

In conclusion, the grammatical approach is suitable for prediction of chemical reactions of the functional group-to-functional group, as well as of the basic mechanism-based reactions. Currently, the design is applied to the reaction of one or two molecules. The application of the term-rewriting grammar in describing more complicated reactions is presented in the following chapter.

REFERENCES

1. Maude System. http://maude.cs.uiuc.edu/
2. Eker, S.; Knapp, M.; Laderoute, K.; Lincoln, P.; Talcott, C. In *Pathway Logic: Executable Models of Biological Networks*, 4th International Workshop on Rewriting Logic and its Applications (WRLA'02). , Italy, 2002; Italy, 2002.
3. http://www.chempensoftware.com/reactions/RXN034.htm

Chapter 6: Application of Term-Rewriting Grammar in Describing Complicated Reactions

A challenge is describing chemical reactions using computers is that they require different information. The previous chapter demonstrated that the term-rewriting grammar Maude could be used to describe the simple reactions of the functional group-to-functional group interactions, as well as of the basic mechanism-based reactions. In this chapter, Maude is used to describe more complicated mechanism-based reactions: acetoacetic ester synthesis, alder ene reaction, and gassman indole synthesis.

6.1 Acetoacetic Ester Synthesis

The first half of the acetoacetic ester synthesis reaction is shown in Figure 6.1, and the atom labeling of the starting compound is shown in Figure 6.2.

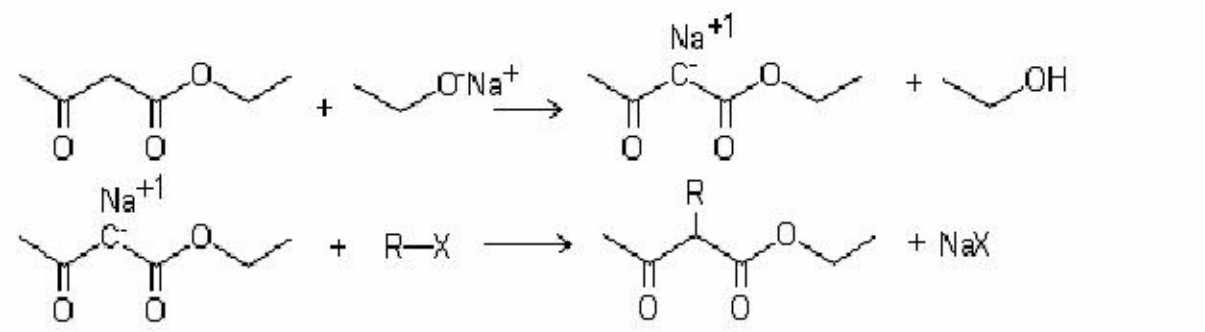


Figure 6.1: Acetoacetic ester synthesis [1]

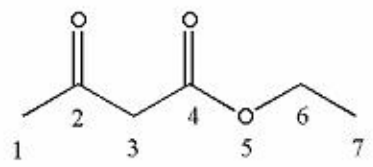


Figure 6.2: Atom labeling of the acetoacetic ester synthesis starting compound

This reaction requires the third carbon (C3-3) to be in between the two carbonyls. The terms describing the starting material for the reaction are (C3-1, C2-2, C3-3, C3-4, O3-5, C3-6, C3-7). Since the C3-3 is the only atom that is involved in this part of the reaction, the rules are developed to focus on the transformation of this atom:

(1) rl C3-3 => | anion-formation system : anion-C3-3 | .

(2) rl anion-C3-3 => | alkyl-halide-substitution system : R-C3-3 | .

These rules are programmed into two different modules in AcetoaceticEsterSynthesis.maude.

```
mod MODULE1 is

sorts Process ChemFeature State Transformation .
subsorts ChemFeature Process State < Transformation .

op _,_ : ChemFeature ChemFeature -> ChemFeature .
ops C3-1 C2-2 C3-3 C2-4 O3-5 C3-6 C3-7 : -> ChemFeature .
op {_ : Process -> Process .
op _} : State -> State .
op atom-activation : -> Process .
ops charged-C3-3 : -> State .
op _:_ : Process State -> Transformation .
op alkyl-halide-substitution : -> Process .
op  R-C3-3 : -> State .

rl C3-3 => { atom-activation : charged-C3-3 } .

endm
mod MODULE2 is
sorts Process ChemFeature State Transformation .
subsorts ChemFeature Process State < Transformation .

op _,_ : ChemFeature ChemFeature -> ChemFeature .
ops C3-1 C2-2 C3-3 C2-4 O3-5 C3-6 C3-7 : -> ChemFeature .
op {_ : Process -> Process .
op _} : State -> State .
op atom-activation : -> Process .
ops charged-C3-3 : -> State .
op _:_ : Process State -> Transformation .
op alkyl-halide-substitution : -> Process .
op  R-C3-3 : -> State .

rl charged-C3-3 => { alkyl-halide-substitution : R-C3-3 } .
endm
```

Figure 6.3 The two modules in AcetoAceticEsterSynthesis.maude. The lines enclosed in the red boxes are the rules describing the two steps in the reaction

The starting material terms are entered into the first module, and the result is shown below:

```
[patra@ariadne maude-linux]$ maude
                     \||||||||||||||||||/
                   --- Welcome to Maude ---
                     /||||||||||||||||||\
            Maude 2.3 built: Feb 14 2007 17:53:50
            Copyright 1997-2007 SRI International
                   Wed Apr  9 03:44:36 2008
Maude> load MODULE1.maude
Warning: "MODULE1.maude", line 56 (mod MODULE3): multiple distinct parses for statement
rl (C3-1, C2-2, R-C3-3, C2-4, O3-5, C3-6, C3-7) => (dilute-acid : (C3-1, C2-2, R-C3-3) + (hydroxyl-O3-5, C3-6, C3-7)) .
Warning: "MODULE1.maude", line 57 (mod MODULE3): multiple distinct parses for statement
rl (R-C2-2, R-C3-3, C2-4, R-O3-5) => (dilute-acid : (R-C2-2, R-C3-3) + (R-hydroxyl-O3-5)) .
Warning: "MODULE1.maude", line 78 (mod MODULE4): multiple distinct parses for statement
rl (C3-1, C2-2, R-C3-3, C2-4, O3-5, C3-6, C3-7) => (alcoholic-alkali : (C3-1, C2-2, OH) + (R-C3-3, C2-4, hydroxyl-O3-5) + (C3-6, C3-7, OH)) .
Maude> rew in MODULE1 : (C3-1 , C2-2 , C3-3 , C2-4 , O3-5 , C3-6 , C3-7) .
Warning: <standard input>, line 2: ambiguous term, two parses are:
((C3-1,(C2-2,C3-3)),(C2-4,O3-5)),(C3-6,C3-7)
-versus-
(((C3-1,C2-2),C3-3),(C2-4,O3-5)),(C3-6,C3-7)

Arbitrarily taking the first as correct.
rewrite in MODULE1 : ((C3-1,(C2-2,C3-3)),(C2-4,O3-5)),(C3-6,C3-7) .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result [Transformation]: ((C3-1,(C2-2,((anion-formation : anion-C3-3)))),(C2-4,O3-5)),(C3-6,C3-7)
```

Figure 6.4: Term-rewriting result from MODULE1 for input (C3-1, C2-2, C3-3, C3-4, O3-5, C3-6, C3-7). The last line is the returned result

The result (the last line) shows that the rule for the anion formation has been applied to the C3-3. This output is processed to remove the process name (anion-formation) and the excess parentheses, such that the output becomes (C3-1, C2-2, anion-C3-3, C2-4, O3-5, C3-6, C3-7). The new output is now an input for the second module, and the result is:

```
Maude> rew in MODULE2 : (C3-1, C2-2, anion-C3-3 , C2-4 , O3-5 , C3-6 , C3-7) .
Warning: <standard input>, line 3: ambiguous term, two parses are:
((C3-1,(C2-2,anion-C3-3)),(C2-4,O3-5)),(C3-6,C3-7)
-versus-
(((C3-1,C2-2),anion-C3-3),(C2-4,O3-5)),(C3-6,C3-7)

Arbitrarily taking the first as correct.
rewrite in MODULE2 : ((C3-1,(C2-2,anion-C3-3)),(C2-4,O3-5)),(C3-6,C3-7) .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result [Transformation]: ((C3-1,(C2-2,((alkyl-halide-substitution : R-C3-3)))),(C2-4,O3-5)),(C3-6,C3-7)
Maude>
```

Figure 6.5: Output from MODULE 2 for (C3-1, C2-2, anion-C3-3, C3-4, O3-5, C3-6, C3-7)

The result indicates that the alkyl-halide-substitution rule has been successfully applied to the input. The result from MODULE2 represents the final product from the first-half reaction. This product, or an intermediate product, serves as a reactant for the second half of the acetoacetic ester synthesis reaction, Figure 6.6.

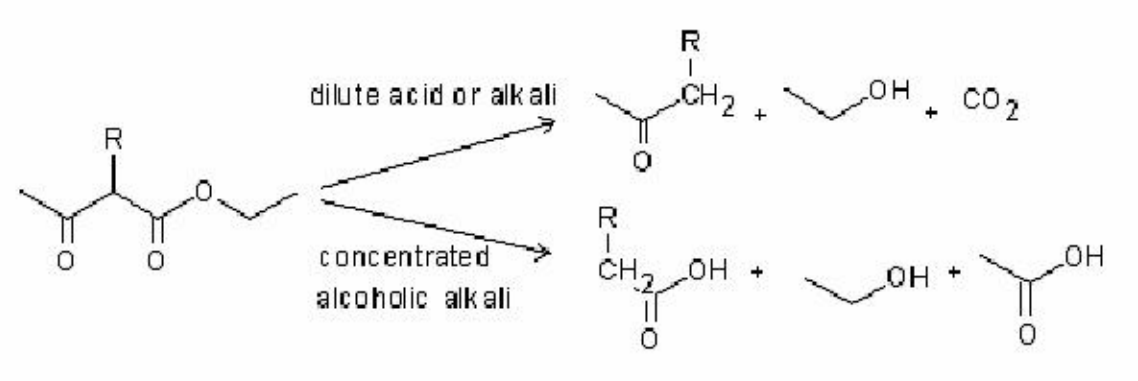
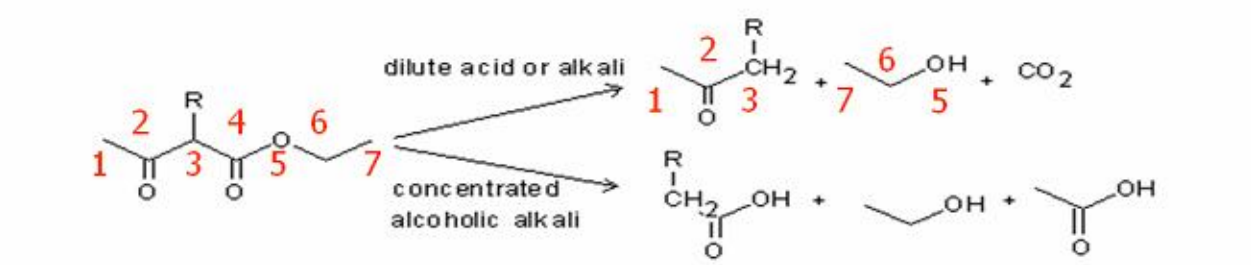


Figure 6.6: The second–half reaction for acetoacetic ester synthesis [1]

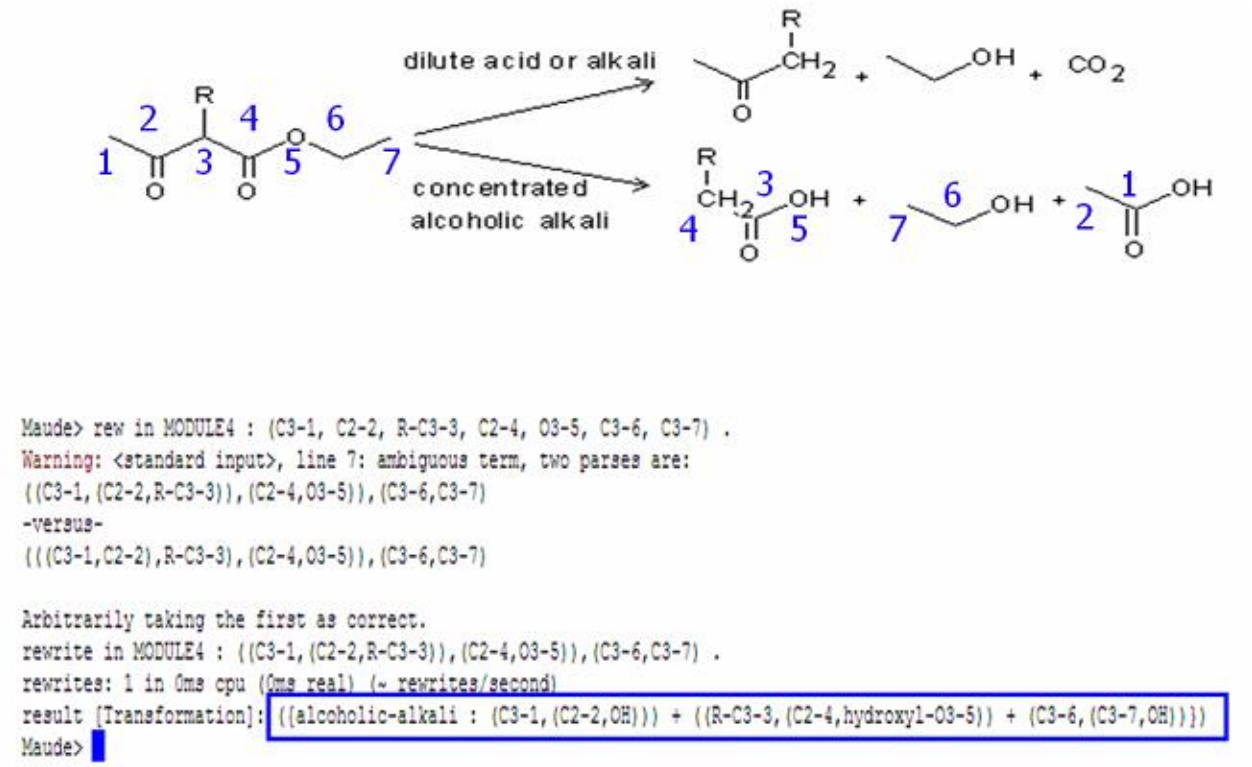
As shown in the pathway, the intermediate product can continue further through either the dilute acid/alkali pathway or through the concentrated alcoholic alkali pathway. The two reactions are programmed into two separated modules, MODULE3 and MODULE4, because they require the same input. Figures 6.7 and 6.8 show the rules that describe the two pathways. The results from both modules indicate that both of the rules have been successfully applied to the input.




```
Maude> rew in MODULE3 : (C3-1, C2-2, R-C3-3, C2-4, O3-5, C3-6, C3-7) .
Warning: <standard input>, line 2: ambiguous term, two parses are:
((C3-1,(C2-2,R-C3-3)),(C2-4,O3-5)),(C3-6,C3-7)
-versus-
(((C3-1,C2-2),R-C3-3),(C2-4,O3-5)),(C3-6,C3-7)

Arbitrarily taking the first as correct.
rewrite in MODULE3 : ((C3-1,(C2-2,R-C3-3)),(C2-4,O3-5)),(C3-6,C3-7) .
rewrites: 1 in 0ms cpu (0ms real) (∞ rewrites/second)
result [Transformation]: {dilute-acid : ((C3-1,(C2-2,R-C3-3)) + (hydroxyl-O3-5,(C3-6,C3-7)))}
Maude>
```

Figure 6.7: The reaction pathway for the intermediate product through the dilute acid or alkali (MODULE3). The numbers in red show the decomposition of the atoms as a result of the reaction, and the information enclosed in the red box is the result returned by the program




```
Maude> rew in MODULE4 : (C3-1, C2-2, R-C3-3, C2-4, O3-5, C3-6, C3-7) .
Warning: <standard input>, line 7: ambiguous term, two parses are:
((C3-1,(C2-2,R-C3-3)),(C2-4,O3-5)),(C3-6,C3-7)
-versus-
(((C3-1,C2-2),R-C3-3),(C2-4,O3-5)),(C3-6,C3-7)

Arbitrarily taking the first as correct.
rewrite in MODULE4 : ((C3-1,(C2-2,R-C3-3)),(C2-4,O3-5)),(C3-6,C3-7) .
rewrites: 1 in 0ms cpu (0ms real) (~ rewrites/second)
result [Transformation]: ((alcoholic-alkali : (C3-1,(C2-2,OH))) + ((R-C3-3,(C2-4,hydroxyl-O3-5)) + (C3-6,(C3-7,OH))))
Maude>
```

Figure 6.8: The reaction pathway for the intermediate product through the concentrated alcoholic alkali (MODULE4). The numbers in blue show the decomposition of the atoms as a result of the reaction, and the information enclosed in the blue box is the result returned by the program

6.2 Alder Ene Reaction

The reaction of Alder ene reaction is a bit more challenging than the acetoacetic ester synthesis.

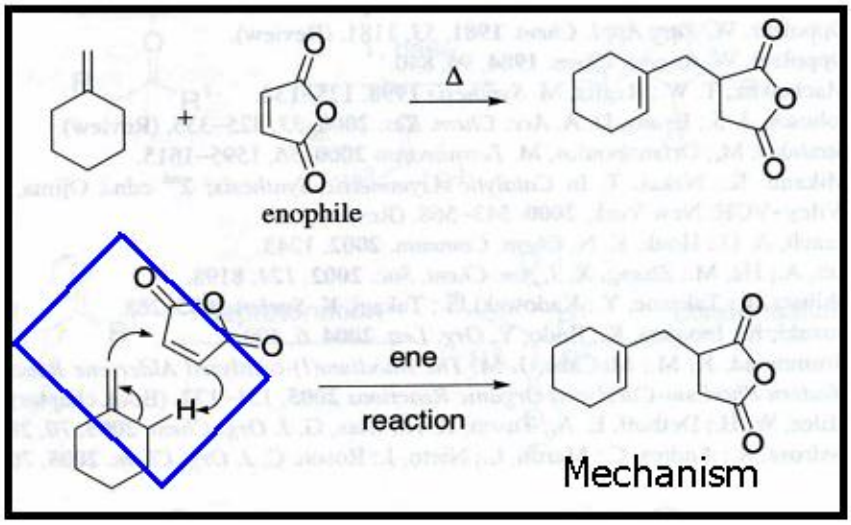


Figure 6.9: Outline reaction for alder ene. The focus of the reaction is enclosed in the blue box [2]

Figure 6.10 shows two example reactions of Alder ene.

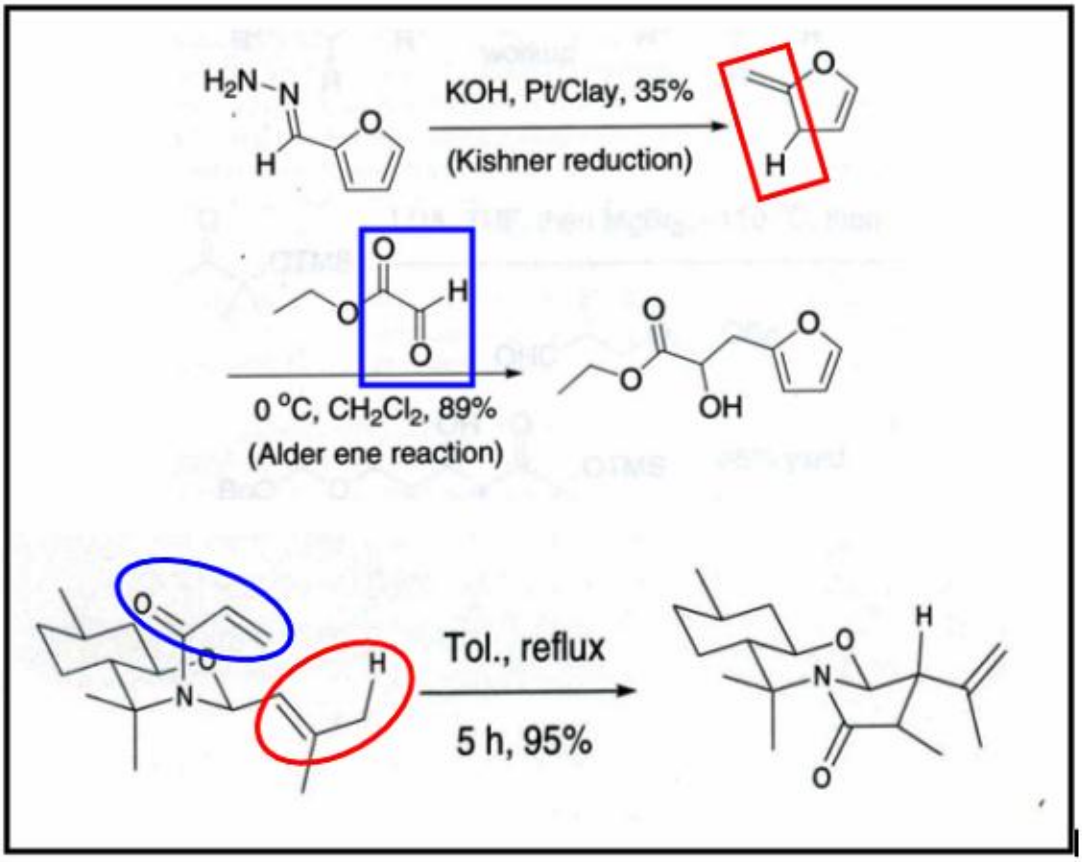
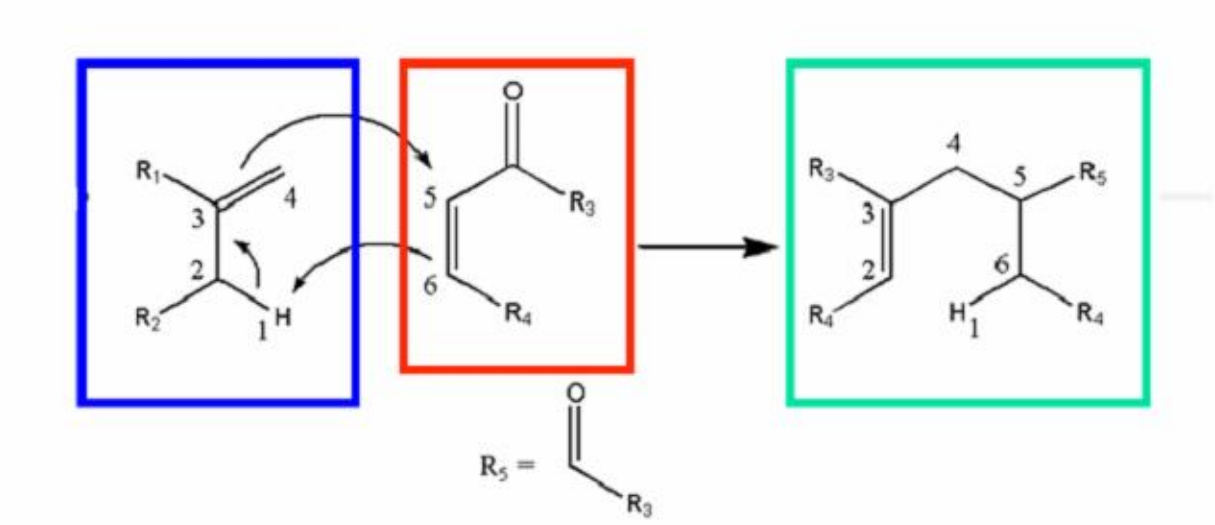


Figure 6.10: Two examples of Alder ene reactions. The reactive portions of the reactants are enclosed in the blue and red boxes [2]

As shown in the examples, an exact match of the starting materials is not crucial. The reaction focuses on the proton at the allylic position to an alkene in one compound, and the conjugate system in another compound.

The Maude program that handles this reaction is divided into three modules.




```
mod ALLYLIC_H is
        rl (allytic_proton) => (H1, R2-C3-2, alkene-C2-3, alkene-C2-4)
endm
```

```
mod CONJUGATE_SYSTEM is
        rl (primary-conjugate-system) => (R5-alkene-C2-5, R4-alkene-C2-6)
endm
```

```
mod REACTION is
        (H1, R2-C3-2, alkene-C2-3, alkene-C2-4) + (R5-alkene-C2-5, R4-alkene-C2-6) =>
                (R2-alkene-C2-2, R3-alkene-C2-3, C3-4, R5-C3-5, R4-H1-C3-6)
endm
```

Figure 6.11: Transformation rules describing the Alder ene reaction. The rules are declared separately in three different modules. The color of the borderline surrounding each module matches the color of the borderline surrounding the fragment that the module describes

Each module describes the transformation of an input term. Each of the input terms corresponds to a starting material for the reaction. For instance, 'allytic_proton' represents the fragment that contains a proton at the allytic position to the alkene (enclosed in the blue box), and 'primary-conjugate-system' represents the fragment containing the conjugated system (enclosed in the red box). The input term are designed to describe the chemistry of the starting material as much as possible. The term 'primary' in 'primary-conjugate-system' differentiates the conjugate system at the primary position from the internal conjugate system. The modules then transform the input terms to their corresponded components. The 'REACTION' module combines the two sets of the transformed terms to a new set of terms that describes the components of the final product. The results when the inputs are entered their modules are shown below:

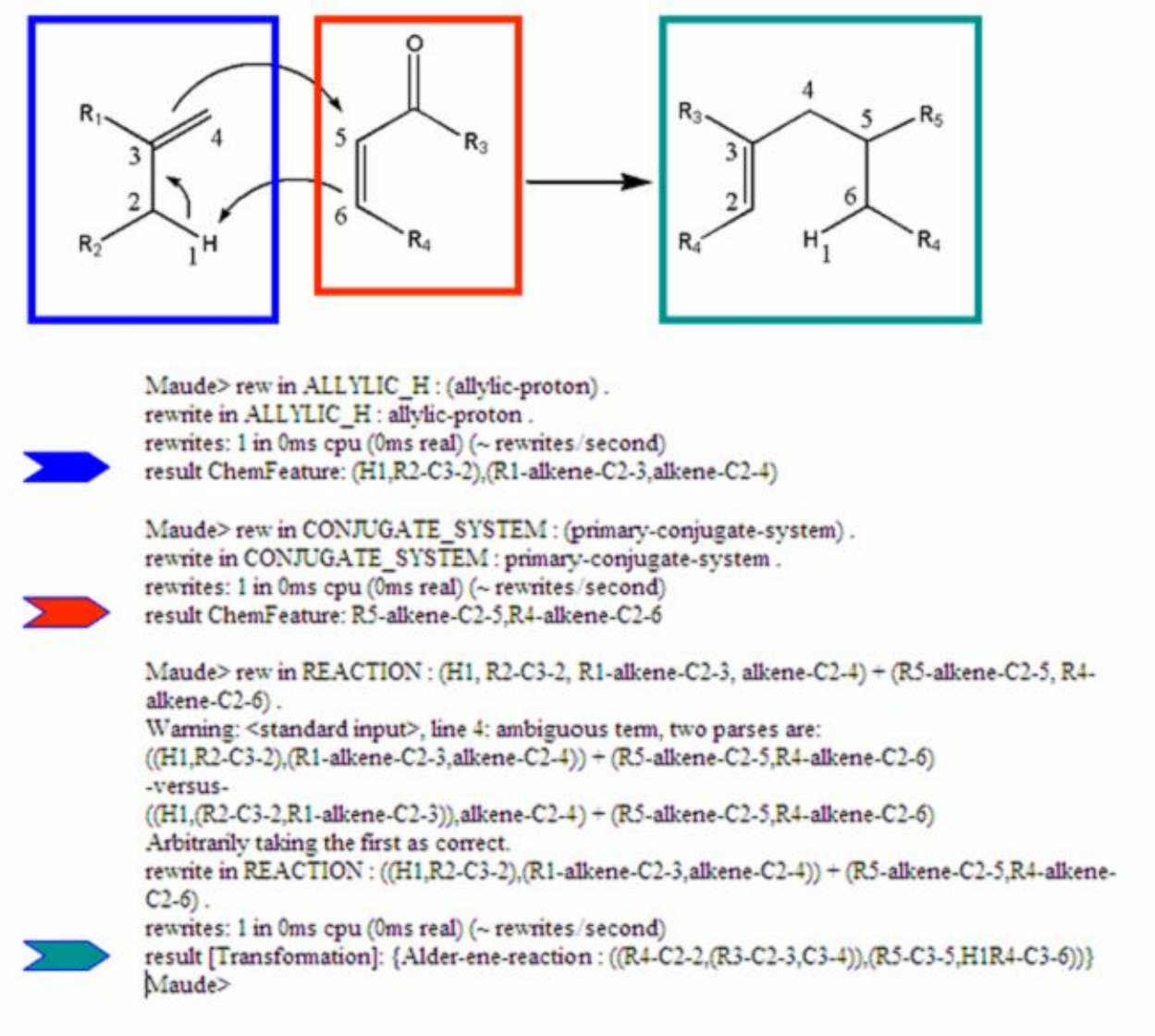


Figure 6.12: Results obtained the three modules of Alder ene reaction. The result from each module is indicated by a colored arrow. These arrows are color coded to match their fragments

6.3 Gassman Indole Synthesis

The reaction of the Gassman Indole synthesis is shown below.

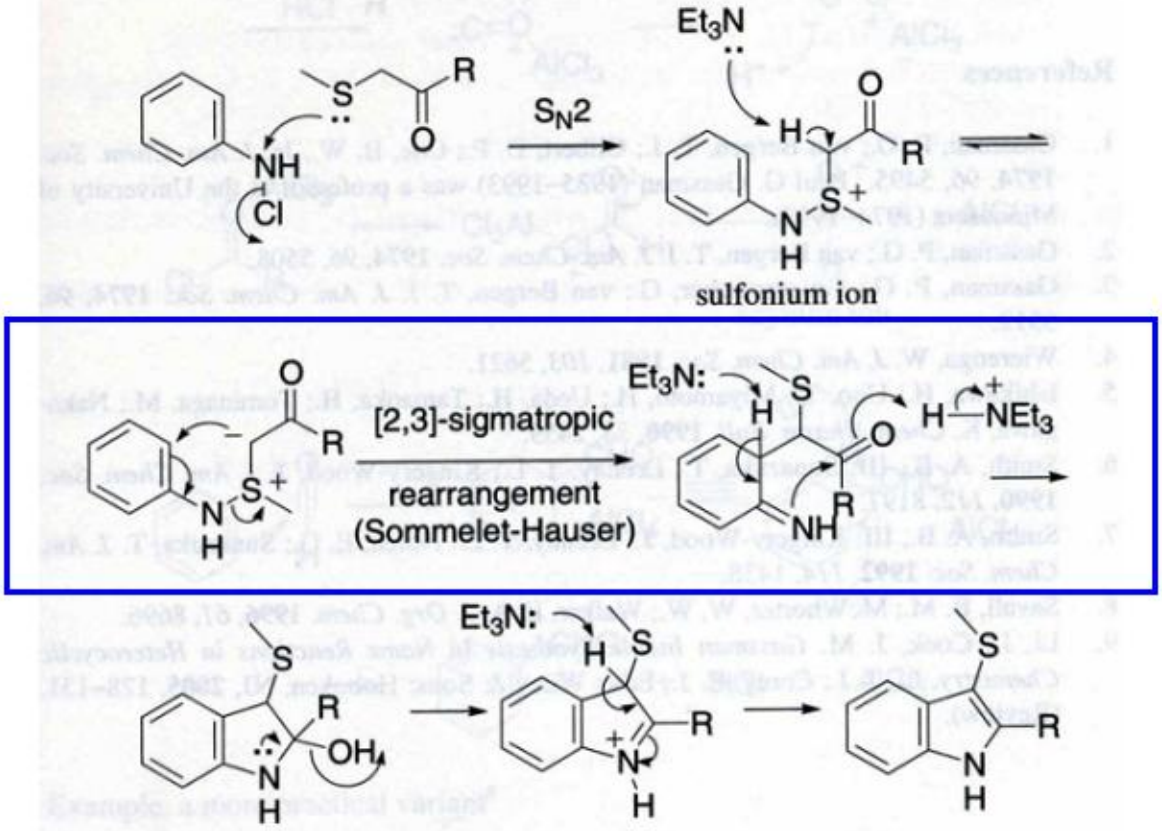
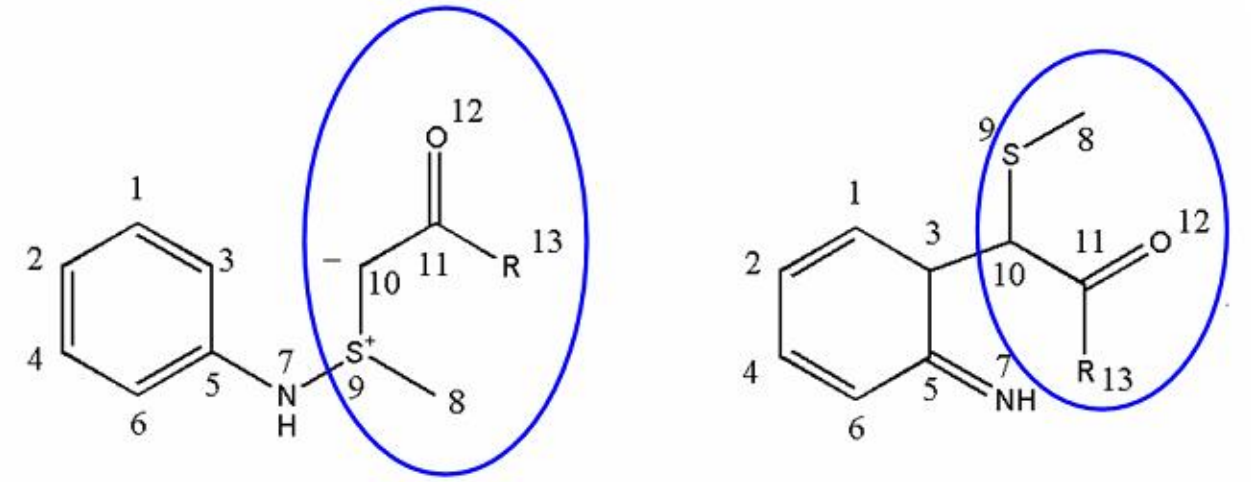


Figure 6.13: Gassman Indole Synthesis [3]

The purpose here is to examine if the term-rewriting can be designed to describe complicate mechanisms, such as the Sommelet-Houser rearrangement in this reaction. The design begins with the intermediate anion structure. The rules that describe the changes in the first step is shown below:



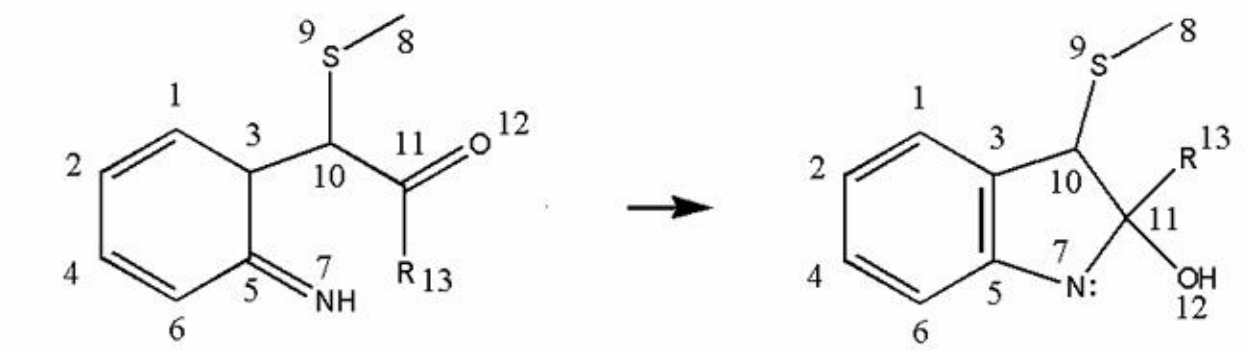
Input = ([Ca-1, Ca-2, Ca-3, Ca-4, Ca-5, Ca-6], N3-7{5}, plus-S34-9,
minus-C3-10, C2-11, O2-12)
rl [Ca-1, Ca-2, Ca-3, Ca-4, Ca-5, Ca-6] = benzene
rl (plus-S34-9, minus-C3-10, C2-11, O2-12) = A
Structure A = (benzene, N3-7{5}, A)
rl benzene = 1,4-cyclohexadien
rl N3-7{5} = N1-7{5}
rl A = A{3}
Structure B = (1,4-cyclohexadien, N1-7{5}, A{3})

Figure 6.14: The rules describing the transformation of the terms describing Structure A to those describe Structure B. The terms printed in blue represent those atoms that are included in the blue circle

The input describing the components of the anion structure (Structure A) is shown in the first line. The atomtypes Ca 1-6 represent the aromatic carbons in the benzene ring. N3-7{5} represents N7 whose atomic geometry is N3 and is attached to the fifth position on the benzene ring. The remaining of the terms describes the atomic geometry of S9, C10, C11, and C12, respectively. R13 is omitted from the rule because it is not involved in the transformation. To shorten the input string, the aromatic atoms enclosed in the brackets are first transformed to a

new term 'benzene'. The remaining of molecule, excluding N7, is also assigned to a new term 'A'. Hence, the new term describe Structure A is now (benzene, N3-7{5}, A). The last three rules describe the shifting of the 'A' fragment. As a result of the shifting, the benzene ring got converted to 1,4-cyclohexadien. In addition, the atomic geometry of N7 changes from sp3 (N3) to sp1 (N1), and the 'A' fragment is now attached to the third position on the ring (A{3}). This yield a resulting term describing Structure B as (1,4-cyclohexadien, N1-7{5}, A{3}).

The next step is to describe the transformation from Structure B to Structure C.



Structure B = (1,4-cyclohexadien, N1-7{5}, A{3}) Structure C

rl 1,4-cyclohexadien = benzene

rl A{3} = (S34-9, C3-10, C2-11, O2-12){3}

rl N1-7{5} = N3-7{5}

New Input: (benzene, N3-7{5}, (S34-9, C3-10, C2-11, O2-12){3})

(benzene, N3-7{5}, (S34-9, C3-10, C2-11, O2-12){3}) = (:indole, S34-9{10}, R-13{11}, O3-12{11})

Structure C = (:indole, S34-9{10}, R-13{11}, O3-12{11})

Figure 6.15: Rules describing the term transformation from Structure B to Structure C. The blue terms in line 5 are those atoms that are involved in the indole structure in Structure C

1,4-cyclohexadien is transformed back to a 'benzene'. The 'A' fragment is also transformed back to its original component representation (S34-9, C3-10, C2-11, O2-12), and the atomic geometry of N7 also changes from sp1 to sp3. These transformations yield an output of

(benzene, N3-7{5}, (S34-9, C3-10, C2-11, O2-12){3}). The next step involves collecting the terms that are involved in the new structure, indole. This yield a new term for Structure C as (:indole, S34-9{10}, R-13{11}, O3-12{11}). The colon in from the of the term 'indole' represents the electrons at the N7.

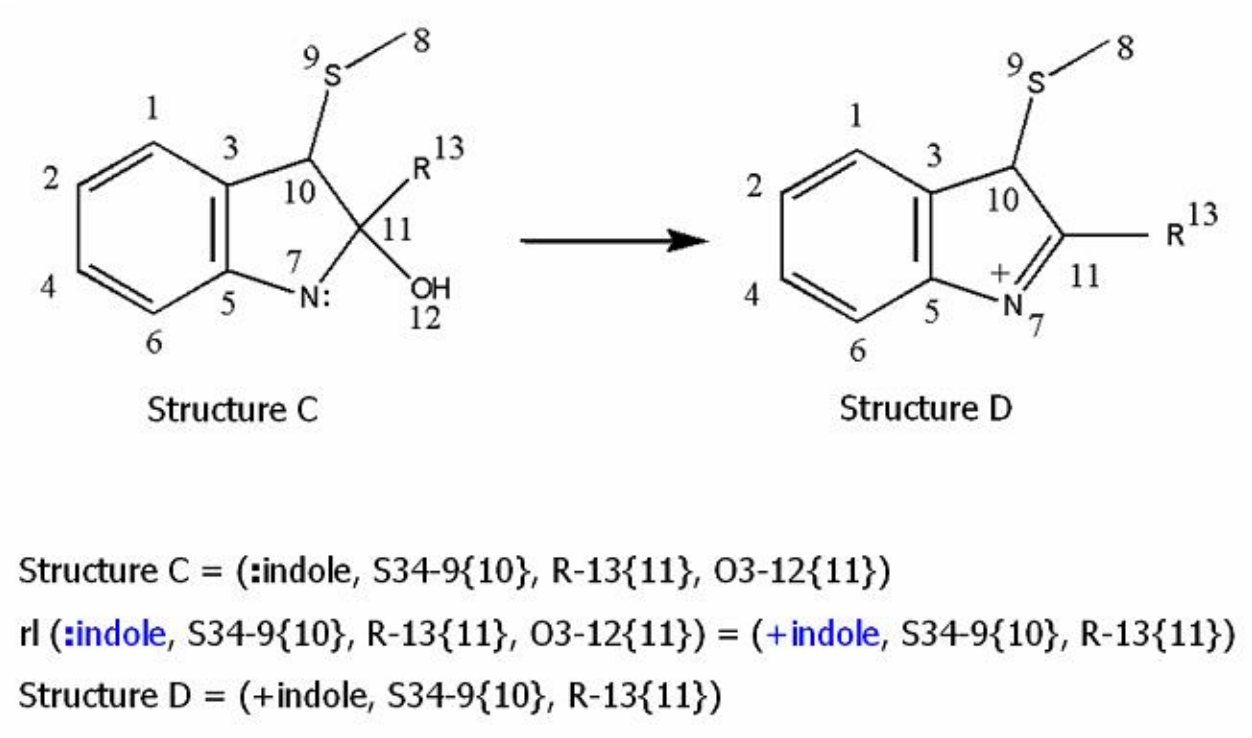
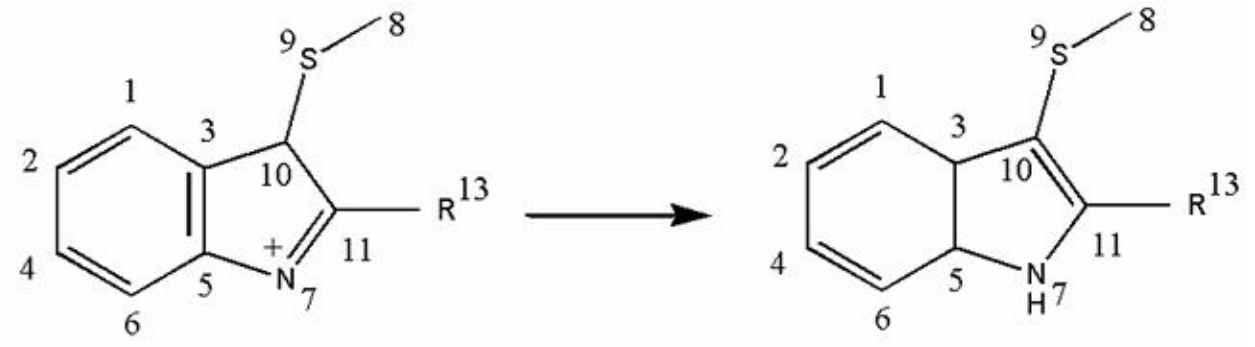


Figure 6.16: Rules describing the term transformation from :indole to a cation indole (+indole)



Structure D = (+indole, S34-9{10}, R-13{11})
rl (+indole, S34-9{10}, R-13{11}) = (indole, S34-9{10}, R-13{11})
Structure E = (indole, S34-9{10}, R-13{11})

Figure 6.17: Rules describing term transformation from +indole and the neutral indole

Figures 6.16 and 6.17 describe the remaining of the transformations for the reactions, which involves the transformation from :indole to +indole (Figure 6.16) and from +indole to the neutral indole (Figure 6.17).

6.4 Discussion

Based on the results observed thus far, it can be concluded that the term-rewriting grammar is also suitable to describe the complicated reactions. The terms and rules can be designed accordingly to satisfy the reactions' requirements. An advantage of the grammatical approach is that it allows a one-to-one mapping of the reaction expressions that are commonly observed in the literatures onto the transformation rules. This bypasses the two challenges encountered with the procedural programming languages: the need of finding a universal representation for the reactions and of defining the required conditions to the computers. The immediate pattern recognition by the grammar can quickly assign the reactions to the functional group patterns in one step, instead of going through a series of step-by-step evaluations as with

the procedural programming. Connectivity information can be included in the rules for those reactions that focus on a small portion of the molecules. However, with advanced and complicated reactions, it is more feasible to ignore the connectivity and focus the transformations more on the chemical term simplification. For instance, in the asman indole synthesis, it is useful and chemically equivalent to assign the six Ca terms to a benzene term. Similarly, in the process of converting Structure B to Structure C (Figure 6.15), it is also chemically equivalent to collect all the atom terms that are involved in the new ring structure and assign them a new term, :indole. This is similar to the collecting of like terms in algebra. With small molecules, the simple functional group-to-functional group reactions can be directly applied. However, for the advanced reactions, the similarity between the patterns embedded in the sample molecules and those in the reaction templates may first need to be determined. This can be performed using several similarity determination methods.

REFERENCES

1. Acetoacetic Ester Synthesis. http://www.chempensoftware.com/reactions/RXN002.htm
2. Li, J. J., Alder ene reaction. In *Name Reactions: A Collection of Detailed Reaction Mechamism (3rd Edition)*, Springer-Verlag Berlin Heidelberg: Germany, 2006; pp 1-2.
3. Li, J. J., Gassman indole synthesis. In *Name Reactions: A Collection of Detailed Reaction Mechamism (3rd Edition)*, Springer-Verlag Berlin Heidelberg: Germany, 2006; pp 257-258.

Chapter 7: Project Conclusion

The overall design of the project is shown in Figure 7.1.

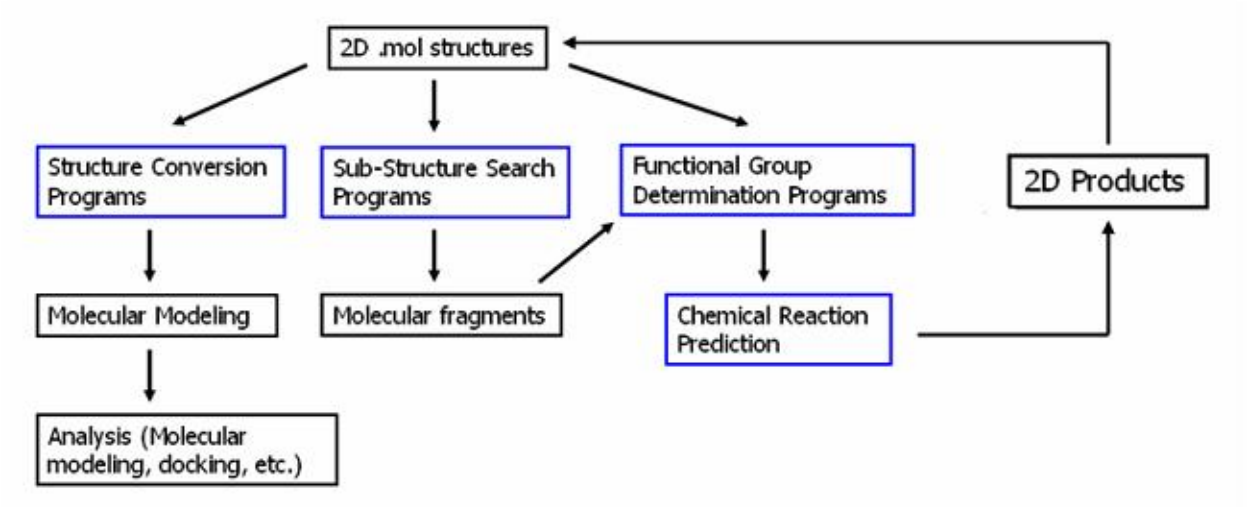


Figure 7.1: Overall design and applications of the algorithms (blue) developed in this project

A set of algorithms have been successfully designed to extract information from the 2D compound representation. These include the chemical property prediction algorithm (the structure conversion programs in Figure 7.1), the sub-structure search algorithms, and the functional groups determination algorithms (which include the functional group reactivity determination algorithm). A set of the term-rewriting programs have also been designed for approximately 30-35 simple reactions and 10 advanced reactions. Overall, this dissertation work has successfully setup a fundamental layout for a chemical design network. Reaction rules of inorganic and metal chemistry would be useful and worth adding to the system's knowledge library. This can be carried out as a part of the future work for the subsequent projects. Another useful future work is to automate the programs. This will allow the system to continuously analyze the 2D structures in the database. The generated new products can, then, be converted to 3D such that their uses in drug or inhibitor design, for examples, can be determined.

8. APPENDICES

Appendix A: List of the Functional Group Search Algorithm Tables

Table 2.1: Search algorithmic procedures for amide

Amide	
Input	.mol structure file
Search Criteria	Search for nitrogen that is single bonded to a carbon that is double-bonded to oxygen
Output	(1) the atom numbers (as appeared in the .mol input file) of the nitrogen, carbon, and oxygen that fit the searched pattern (2) total number of active bonds

Table 2.2: Search algorithmic procedures for acid

Acid	
Input	.mol structure file
Search Criteria	(1) Search for carbon atom that is single bonded to an oxygen (O1) and double-bonded to another oxygen (O2) (2) Verify that O1 only has one single-bonded neighbor, and it must be the carbon
Output	(1) the atom numbers (as appeared in the .mol input file) of the carbon, and the two oxygen atoms **(2)** total number of active bonds

Table 2.3: Search algorithmic procedures for ester

Ester	
Input	.mol structure file
Search Criteria	(1) Search for carbon atom that is single bonded to an oxygen (O1) and double-bonded to another oxygen (O2) (2) Verify that O1 has two single-bonded neighbors, and one must be the carbon
Output	(1) the atom numbers (as appeared in the .mol input file) of the carbon, and the two oxygen atoms **(2)** total number of active bonds

Table 2.4: Search algorithmic procedures for amine

Amine	
Input	.mol structure file
Search Criteria	Search for nitrogen atom that only has single-bonded neighbors
Output	(1) the atom numbers (as appeared in the .mol input file) of the nitrogen **(2)** total number of active bonds

Table 2.5: Search algorithmic procedures for alcohol

Alcohol	
Input	.mol structure file
Search Criteria	Search for oxygen atom that only has one single-bonded neighbor
Output	(1) the atom numbers (as appeared in the .mol input file) of the oxygen **(2)** total number of active bonds

Table 2.6: Search algorithmic procedures for ether

Ether	
Input	.mol structure file
Search Criteria	Search for nitrogen atom that has two single-bonded neighbors
Output	the atom numbers (as appeared in the .mol input file) of the oxygen

Table 2.7: Search algorithmic procedures for alkene

Alkene	
Input	.mol structure file
Search Criteria	Search for carbon that is double-bonded to another carbon
Output	(1) the atom numbers (as appeared in the .mol input file) of the two carbon atoms **(2)** total number of active bonds

Table 2.8: Search algorithmic procedures for alkyne

Alkyne	
Input	.mol structure file
Search Criteria	Search for carbon that is triple-bonded to another carbon
Output	(1) the atom numbers (as appeared in the .mol input file) of the two carbon atoms **(2)** total number of active bonds

Table 2.9: Search algorithmic procedures for thiol

Thiol	
Input	.mol structure file
Search Criteria	Search for sulfur atom that only has single-bonded neighbors
Output	(1) the atom numbers (as appeared in the .mol input file) of the sulfur **(2)** total number of active bonds

Table 2.10: Search algorithmic procedures for nitrile

Nitrile	
Input	.mol structure file
Search Criteria	Search for nitrogen atom that has a triple-bonded to a carbon atom
Output	(1) the atom numbers (as appeared in the .mol input file) of the nitrogen **(2)** total number of active bonds

Table 2.11: Search algorithmic procedures for ketone

Ketone	
Input	.mol structure file
Search Criteria	Search for carbon atom that is double bonded to an oxygen atom and single bonded to two carbon atoms
Output	(1) the atom numbers (as appeared in the .mol input file) of the carbon and the oxygen atoms **(2)** total number of active bonds

Table 2.12: Search algorithmic procedures for aldehyde

Aldehyde	
Input	.mol structure file
Search Criteria	(1) Search for carbon atom that is double bonded to an oxygen atom and single bonded to one carbon atom, OR (2) Search for carbon atom that is double bonded to an oxygen atom and has one single bond to one carbon atom and another single bond to a hydrogen atom
Output	(1) the atom numbers (as appeared in the .mol input file) of the carbon and the oxygen atoms **(2)** total number of active bonds

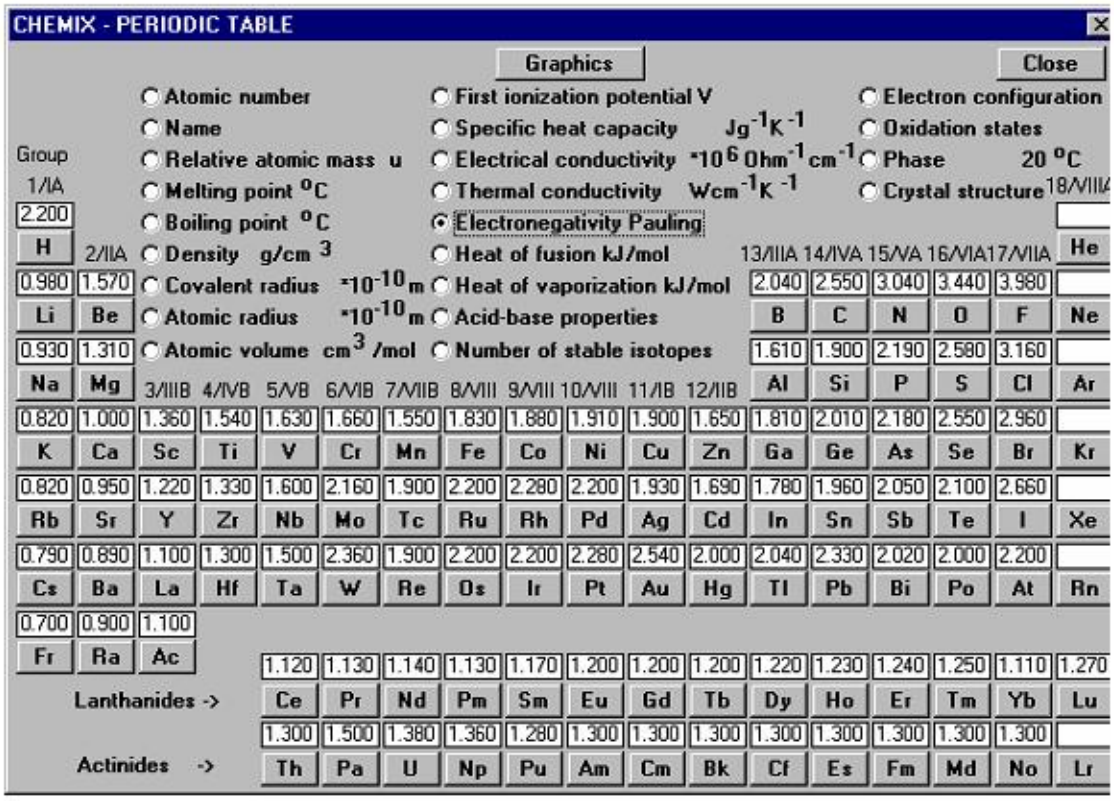
Table 2.13: Search algorithmic procedures for benzene

Benzene	
Input	.mol structure file
Search Criteria	Part I: For isolated benzene rings: (1) Search for carbon, or target carbon (TC), that has a single-bonded carbon neighbor (SBCN_1) and a double-bonded carbon neighbor (DBCN_1) (2) If SBCN_1 is found, for SBCN_1, search (excluding TC) for a double-bonded carbon neighbor (DBCN_2) (3) If DBCN_1 is found, for DBCN_1, search (excluding TC) for a single-bonded carbon neighbor (SBCN_2) (4) If both SBCN_2 and DBCN_2 are found, for SBCN_2, search (excluding DBCN_1) for all double-bonded carbon neighbors (DBCN_3) (5) If both SBCN_2 and DBCN_2 are found, for DBCN_2, search (excluding SBCN_1) for all single-bonded carbon neighbors (SBCN_3) (6) Compare members in both SBCN_3 and DBCN_3 (7) If there is a common member (CM) that exist in both SBCN_3 and DBCN_3, then TC, SBCN_1, DBCN_1, SBCN_2, DBCN_2, SBCN_3, and CM are members of a benzene ring (8) Verify that the members are not already existed (9) Update the overall number of benzene rings and their members that are present in the molecule. Make all bonds in all the rings double-bonds

Table 2.13 (Continued): Search algorithmic procedures for benzene

Benzene	
Search Criteria	Part II: For the remaining of possible ring representation: (1) For each ring that has been found in Part I, search carbon that is jointed between two rings. This carbon is referred to as a joint-carbon (JC) (2) For each JC, search for two carbon atoms that are ring members, with each has at least one single-bonded non-ring neighbor (NRN_1 and NRN_2). The two ring carbon atoms are referred to as ring-member1 (RM1) and ring-member2 (RM2) (3) Find all carbon neighbors, excluding their parental atoms, for both NRN_1 and NRN_2. The neighbors can only be: (a) single-bonded to NRN_1 and double-bonded to NRN_2 or (b) double-bonded to NRN_1 and single-bonded to NRN_2 (4) If the two sets of neighbors contain at least one common carbon, then the JC, RM1, RM2, NRN_1, NRN_2, and the common carbon are members of a ring (5) Verify that the members have not already existed (6) Update the overall ring contents and their members for the molecule.
Output	(1) a list of all rings with their ring number and members (via their atom numbers as appeared in the .mol input file) **(2)** total number of active bonds

Appendix B: Electronegativity Chart [1]



[1] http://www.standnes.no/chemix/periodictable/electronegativity-chart.htm

Appendix C: Detail coding in Ester.maude

```
mod ESTER1 is

sorts Generation ChemFeature Process Compound FunctGroup Name .
subsorts Process < ChemFeature .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> ChemFeature .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops hydrolysis reduction combination : -> Process .
ops system acid alcohol ester amine amide : -> ChemFeature .

*** Rule Rewriting Section

rl { system = ester } => | hydrolysis system : acid | .

endm

-----------------------------------------------------------------

mod ESTER2 is

sorts Generation ChemFeature Process Compound FunctGroup Name .
subsorts Process < ChemFeature .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> ChemFeature .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops hydrolysis reduction combination : -> Process .
ops system acid alcohol ester amine amide : -> ChemFeature .

*** Rule Rewriting Section

rl { system = ester } => | reduction system : alcohol | .
endm
-----------------------------------------------------------------
```

```
mod ESTER3 is

sorts Generation ChemFeature Process Compound FunctGroup Name .
subsorts Process < ChemFeature .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> ChemFeature .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops hydrolysis reduction combination : -> Process .
ops system acid alcohol ester amine amide : -> ChemFeature .

*** Rule Rewriting Section

rl { system = ester + amine } => | combination system : amide | .
rl { system = ester + alcohol } => | combination system : ester | .

endm
```

Appendix D: Detailed Coding in Acid.maude

```
mod ACID1 is

sorts Generation ChemFeature Process Reaction Compound FunctGroup Name .
subsorts Process ChemFeature < Reaction .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> Reaction .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops reduction alkylation dehydration combination : -> Process .
ops system acid alcohol ester amine amide primary-alcohol aldehyde ketone : -> ChemFeature .

*** Rule Rewriting Section

rl { system = acid + alcohol } => | dehydration system : ester | .
rl { system = acid + amine } => | combination  system : amide | .

endm

-------------------------------------

mod ACID2 is

sorts Generation ChemFeature Process Reaction Compound FunctGroup Name .
subsorts Process ChemFeature < Reaction .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> Reaction .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops reduction alkylation dehydration combination : -> Process .
ops system acid alcohol ester amine amide primary-alcohol aldehyde ketone : -> ChemFeature .

*** Rule Rewriting Section

rl { system = acid } => | reduction system : aldehyde | .
endm
```

```
-------------------------------------

mod ACID3 is

sorts Generation ChemFeature Process Reaction Compound FunctGroup Name .
subsorts Process ChemFeature < Reaction .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> Reaction .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops reduction alkylation dehydration combination : -> Process .
ops system acid alcohol ester amine amide primary-alcohol aldehyde ketone : -> ChemFeature .

*** Rule Rewriting Section

rl { system = acid } => | reduction system : primary-alcohol | .
rl { system = acid } => | alkylation system : ketone | .

endm
-------------------------------------

mod ACID4 is

sorts Generation ChemFeature Process Reaction Compound FunctGroup Name .
subsorts Process ChemFeature < Reaction .

op _+_ : ChemFeature ChemFeature -> ChemFeature .
op _,_ : ChemFeature ChemFeature -> ChemFeature .
op _/_ : ChemFeature ChemFeature -> ChemFeature .
op _-_ : ChemFeature ChemFeature -> ChemFeature .
op |__:_| : Process ChemFeature ChemFeature -> Reaction .
op _=_ : ChemFeature ChemFeature -> ChemFeature .
op [_] : ChemFeature -> ChemFeature .
op {_} : ChemFeature -> ChemFeature .
ops reduction alkylation dehydration combination : -> Process .
ops system acid alcohol ester amine amide primary-alcohol aldehyde ketone : -> ChemFeature .

*** Rule Rewriting Section

rl { system = acid } => | alkylation system : ketone | .
endm
```